Título: Instalación y configuración de un cluster de computación

Autor: Santiago Melón Rial

Fecha: 19 de junio de 2012

Titulación: Ingeniería Técnica en Informática de Sistemas

Director: Luis Domingo Velasco

Índice

1 Introducción

1.1 Qué es un cluster

El término cluster se aplica a los conjuntos o conglomerados de computadoras construidos mediante la utilización de componentes de hardware comunes y que se comportan como si fuesen una única computadora.

Hoy en día desempeñan un papel importante en la solución de problemas de las ciencias, las ingenierías y del comercio moderno.

La tecnología de clusters ha evolucionado en apoyo de actividades que van desde aplicaciones de supercómputo y software de misiones críticas, servidores web y comercio electrónico, hasta bases de datos de alto rendimiento, entre otros usos.

El cómputo con clusters surge como resultado de la convergencia de varias tendencias actuales que incluyen la disponibilidad de microprocesadores económicos de alto rendimiento y redes de alta velocidad, el desarrollo de herramientas de software para cómputo distribuido de alto rendimiento, así como la creciente necesidad de potencia computacional para aplicaciones que la requieran.

Simplemente, un cluster es un grupo de múltiples ordenadores unidos mediante una res de alta velocidad, de tal forma que el conjunto es visto como un único ordenador, más potente que los comunes de escritorio.
Los clusters son usualmente empleados para mejorar el rendimiento y/o la disponibilidad por encima de la que es provista por un solo computador, típicamente siendo más económico que computadores individuales de rapidez y disponibilidad comparables.

La construcción de los ordenadores del cluster es más fácil y económica debido a su flexibilidad: pueden tener todos la misma configuración de hardware y sistema operativo (cluster homogéneo), diferente rendimiento pero con arquitecturas y sistemas operativos similares (cluster semi-homogéneo), o tener diferente hardware y sistema operativo (cluster heterogéneo), lo que hace más fácil y económica su construcción.

Para que un cluster funcione como tal, no basta sólo con conectar entre sí los ordenadores, sino que es necesario proveer un sistema de manejo del cluster, el cual se encargue de interactuar con el usuario y los procesos que corren en él para optimizar el funcionamiento.

1.2 Propósitos de un cluster

De un cluster se espera que presente combinaciones de Alto rendimiento, Alta disponibilidad, Balanceo de carga, Escalabilidad).

1.2.1 Alto rendimiento:

Un **cluster de alto rendimiento** es un conjunto de ordenadores que está diseñado para dar altas prestaciones en cuanto a capacidad de cálculo. Los motivos para utilizar un cluster de alto rendimiento son:

- → El tamaño del problema por resolver
- → El precio de la máquina necesaria para resolverlo.

Por medio de un cluster se pueden conseguir capacidades de cálculo superiores a las de un ordenador más caro que el costo conjunto de los ordenadores del clúster.

Ejemplo de clusters baratísimos son los que se están realizando en algunas universidades con computadoras personales desechados por "anticuados" que consiguen competir en capacidad de cálculo con superordenadores carísimos.

1.2.2 Alta disponibilidad:

Un ***cluster de alta disponibilidad*** es un conjunto de dos o más máquinas que se caracterizan por mantener una serie de servicios compartidos y por estar constantemente monitorizándose entre sí. Podemos dividirlo en dos clases:

- → ***Alta disponibilidad de infraestructura***: Si se produce un fallo de hardware en alguna de las máquinas del cluster, el software de alta disponibilidad es capaz de arrancar automáticamente los servicios en cualquiera de las otras máquinas del cluster (failover). Y cuando la máquina que ha fallado se recupera, los servicios son nuevamente migrados a la máquina original (failback). Esta capacidad de recuperación automática de servicios nos garantiza la alta disponibilidad de los servicios ofrecidos por el cluster, minimizando así la percepción del fallo por parte de los usuarios.

- → ***Alta disponibilidad de aplicación***: Si se produce un fallo del hardware o de las aplicaciones de alguna de las máquinas del cluster, el software de alta disponibilidad es capaz de arrancar automáticamente los servicios que han fallado en cualquiera de las otras máquinas del cluster. Y cuando la máquina que ha fallado se recupera, los servicios son nuevamente migrados a la máquina original. Esta capacidad de recuperación automática de servicios nos garantiza la integridad de la información, ya que no hay pérdida de datos, y además evita molestias a los usuarios, que no tienen por qué notar que se ha producido un problema.

1.2.3 Balanceo de carga:

El balanceo de carga es un concepto usado en informática que se refiere a la técnica usada para compartir el trabajo a realizar entre varios procesos, ordenadores, discos u otros recursos. Está íntimamente ligado a los sistemas de multiprocesamiento, o que hacen uso de más de una unidad de procesamiento para realizar labores útiles.

El balanceo de carga se mantiene gracias a un algoritmo que divide de la manera más equitativa posible el trabajo, para evitar los así denominados cuellos de botella.

Un ***cluster de balanceo de carga*** o de ***cómputo adaptativo*** está compuesto por uno o más ordenadores (llamados nodos) que actúan como frontend del cluster, y que se ocupan de repartir las peticiones de servicio que reciba el cluster, a otros ordenadores del cluster que forman el back-end de éste. Un tipo concreto de cluster cuya función es repartir la carga de proceso entre los nodos en lugar de los servicios es el cluster openMosix.

Las características más destacadas de este tipo de cluster son:

→ ***Escalabilidad***: Se puede ampliar su capacidad fácilmente añadiendo más ordenadores al cluster.

→ ***Robustez***. Ante la caída de alguno de los ordenadores del cluster el servicio se puede ver mermado, pero mientras haya ordenadores en funcionamiento, éstos seguirán dando servicio.

1.2.4 Escalabilidad

En telecomunicaciones y en ingeniería informática, la escalabilidad es la propiedad deseable de un sistema, una red o un proceso, que indica su habilidad para extender el margen de operaciones sin perder calidad, o bien manejar el crecimiento continuo de trabajo de manera fluida, o bien para estar preparado para hacerse más grande sin perder calidad en los servicios ofrecidos.

En general, también se podría definir como la capacidad del sistema informático de cambiar su tamaño o configuración para adaptarse a las circunstancias cambiantes. Por ejemplo, una Universidad que establece una red de usuarios por Internet para un edificio de docentes y no solamente quiere que su sistema informático tenga capacidad para acoger a los actuales clientes que son todos profesores, sino también a los clientes que pueda tener en el futuro dado que hay profesores visitantes que requieren de la red por algunos aplicativos académicos, para esto es necesario implementar soluciones que permitan el crecimiento de la red sin que la posibilidad de su uso y re-uso, disminuya o que pueda cambiar su configuración si es necesario.

La escalabilidad como propiedad de los sistemas es generalmente difícil de definir en cualquier caso, en particular es necesario definir los requerimientos específicos para la escalabilidad en esas dimensiones donde se crea que son importantes. Es una edición altamente significativa en sistemas electrónicos, bases de datos, ruteadores y redes. A un sistema cuyo rendimiento es mejorado después de haberle añadido más capacidad

hardware, proporcionalmente a la capacidad añadida, se dice que pasa a ser "un sistema escalable"

1.3 Gestión de Recursos Distribuidos: Sistemas Gestores de Colas

Los sistemas de gestión de colas, gestionan una cola de ejecución, planifican la ejecución de las tareas y gestionan los recursos, para minimizar costes y maximizar rendimiento de las aplicaciones.
Funcionamiento:
1. Los usuarios envían trabajos.

2. El gestor de recursos registra el trabajo.

3. Tan pronto los recursos pedidos se hallen disponibles, el gestor de colas pone a ejecución el trabajo solicitado que según su planificación es el que tiene mayor prioridad.

4. Se puede consultar el estado de los trabajos, en ejecución, en espera o terminados.

5. Se puede eliminar un trabajo.

6. El gestor de colas es configurable.

1.4 El solver

Un solver es un término genérico que se refiere a una pieza de software, posiblemente en la forma de un programa de ordenador independiente, o de una librería, que “resuelve” un problema matemático (teoremas, ecuaciones lineales y no lineales, árboles de expansión, algoritmos de búsqueda…). Un solver toma como entrada la descripción del problema de una manera específica, y calcula su solución, que da como salida. En un solver, se hace hincapié en la creación de un programa o librería que pueda ser fácilmente aplicada a otros problemas similares.

2. Descripción y objetivos del proyecto común Infraestructura de Servicios de computación

Con el fin de un mejor entendimiento, es importante, en primer lugar, la definición de dos entidades básicas:

→***Job***: Cada uno de los problemas a resolver.

→***Queue***: Cada una de las colas en donde los jobs se almacenan.

El proyecto está compuesto por varias partes o módulos diferenciados:

→***Portal web***: El front-end para gestionar el proyecto.

→***ClusterManager***: El "core" del proyecto, donde reside toda la inteligencia.

→***DB***: La base de datos.

→***PendingInfo***: La aplicación que avisará al ***clusterManager*** de las modificaciones en la ***DB***.

→***Agent***: La aplicación que resuelve los jobs. El proyecto sólo abarca la parte de comunicaciones del agent.

Este proyecto engloba tres proyectos de final de carrera:

→Portal de servicios de computación (no detallado en esta documentación)

→Cluster de servicios de computación (no detallado en esta documentación)

→Instalacion y configuración de un cluster de computación.

El proyecto global debe servir para gestionar el cluster, desde el front-end que permitirá al administrador gestionar el cluster y a los usuarios demandar los servicios del mismo para resolver jobs, hasta la implementación del gestor de colas y las comunicaciones con los agentes.

De los servicios esperados de un cluster (Alto rendimiento, Alta disponibilidad, Balanceo de carga, Escalabilidad), se detalla a continuación muy brevemente cómo el proyecto los aborda.

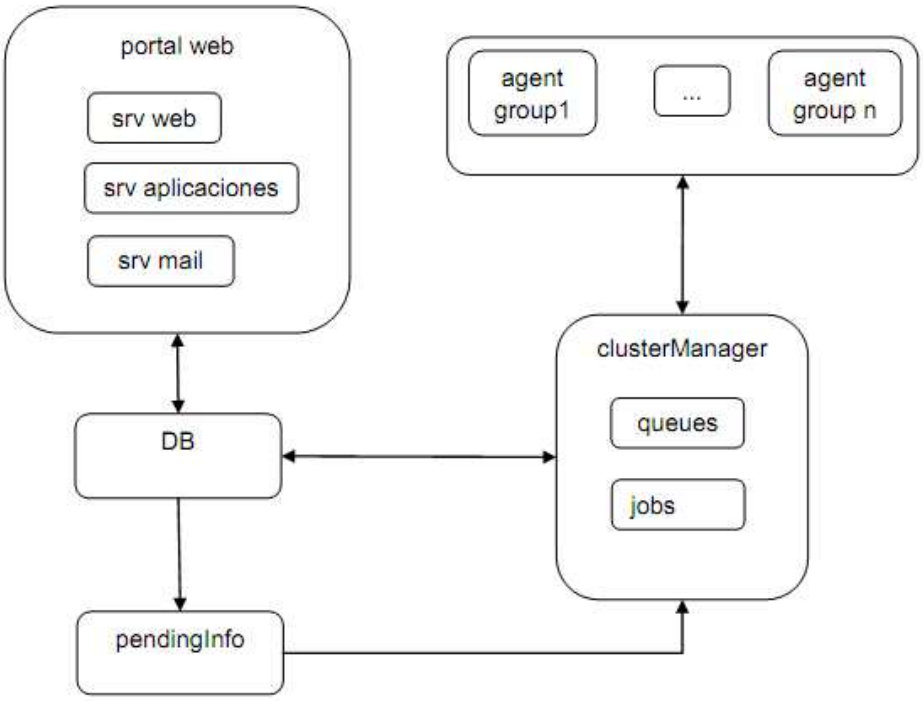
→***Alto rendimiento***: La productividad de la suma de los diferentes agents del cluster ofrece esta prestación.

→***Alta disponibilidad***: al disponer de varios agentes, si uno o varios de ellos caen, el cluster continúa trabajando y resolviendo jobs.

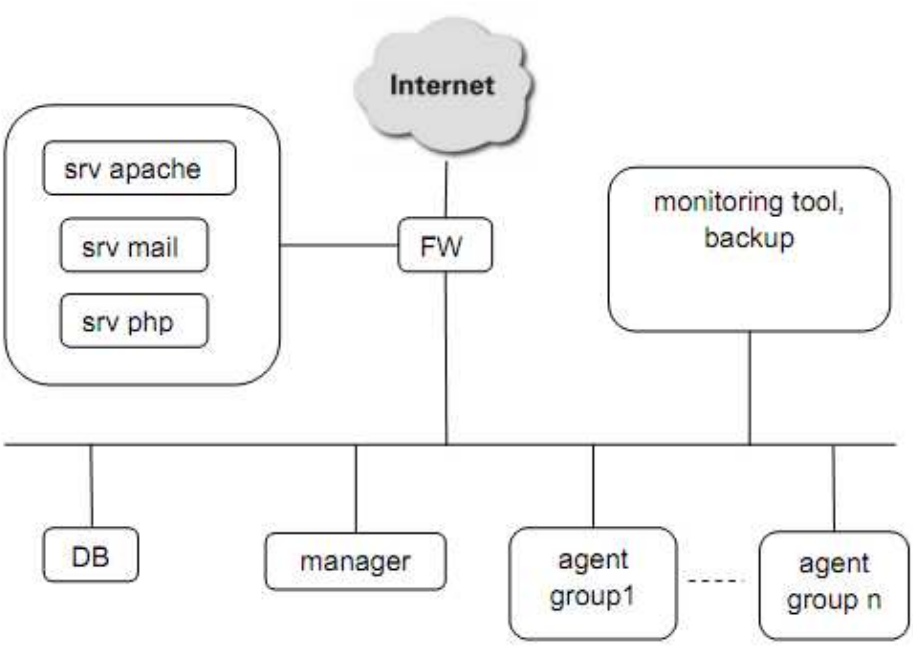
→***Balanceo de carga***: La distribución de los jobs en las diferentes queues, el hecho de que las asociaciones entre agents y queues sea completamente configurable, y el gestor de colas se dan solución a este aspecto.

→***Escalabilidad***: El número de agents es completamente configurable; es posible añadir tantos agents al cluster como sea necesario.

2.1 Esquema conceptual



2.2 Esquema físico de red



2.3 Objetivos

→Diseño e implementación DB

- La DB debe comunicar al cluster cuándo hay nueva información que procesar (creaciones de jobs, de queues, agents, etc...)

→Desarrollar un portal web que gestione el cluster.

- Diseño y planificación del portal web

  · Implementación de la web corporativa, de acceso público y no restringido

- Registro y acceso privado para usuarios web
  · Usuarios

  · Administrador

- Alta y gestión de proyectos

- Alta y gestión de ficheros (datos + algoritmos)

- Alta y gestión de jobs

- Alta y gestión de agentes

- Alta y gestión de colas

- Gestión/visualización de facturación

→Desarrollar un cluster que negocie la gestión de colas para la resolución de problemas por los agentes (que resuelven los problemas accediendo a un solver).

- Diseño e implementación del gestor del cluster

  · Agents, manager, queues, jobs…

  · Gestor de colas

- Diseño e implementación de **simulación de red** física de ordenadores.

- Implementación protocolo de comunicaciones

  · Manager – Agentes

  · DB - Manager

- Generación de facturación de jobs.

3. Descripción y objetivos del proyecto Instalación y configuración de un cluster de computación

3.1 Descripción

El proyecto a realizar consiste en la instalación y configuración de un clúster de computación.
En concreto el clúster dispondrá de:

a) Una zona desmilitarizada que contendrá un servidor web desde el cual se podrán hacer las peticiones externas para acceder al servicio.
b) Un servidor de BBDD y de una subred configurada con DHCP donde se alojarán las diferentes máquinas encargadas de procesar las peticiones.
c) Un firewall entre la red interna y la red externa (internet).
d) Cada parte del sistema dispondrá de un rango de direcciones IP diferente.

3.2 Objetivos

El desarrollo del proyecto contempla las siguientes fases:

1. Instalación de las máquinas.
   Instalación de los sistemas operativos así como todos los controladores necesarios para la puesta a punto de las máquinas implicadas en el proyecto.

2. Instalación de software especifico.
   Instalación de todas las aplicaciones necesarias para la administración de los recursos. Aplicaciones necesarias para montar el servidor web, las bbdd, configuración del firewall (funcionamiento de la red). Así como todo el software extra para el funcionamiento del cluster

3. Configuración de las máquinas.
   Configurar las diferentes aplicaciones de cada máquina.

4. Configuración de la red.
   Configurar la red, asignación de rangos de ip’s comunicación entre la red interna y externa, configurar la DMZ (zona desmilitarizada).

4. La red

4.1 Qué es una red

En informática se denomina **red** al conjunto de máquinas (ordenadores) y software conectados mediante dispositivos (routers, discos, impresoras, etc.) que permiten compartir información, recursos y ofrecer servicios.

También forman parte de la red los diferentes medios de transmisión que permiten enviar y recibir la información necesaria para establecer comunicación entre las diferentes máquinas. Los medios de transmisión suelen ser cables u ondas electromagnéticas (wifi, satélite, etc.) por los que circula la información.

Las redes permiten que los recursos y la información sean accesibles desde cualquier parte sin la necesidad de establecer una conexión directa entre los diferentes computadores y dispositivos que contienen dichos recursos.

Dentro de una red se pueden hacer subdivisiones llamadas **subredes**. Realizando estás subdivisiones se pretende conseguir una mayor organización ya sea para conseguir mayor eficiencia, seguridad o simplemente para agrupar un conjunto de máquinas que deben poder comunicarse entre ellas.

Si nos fijamos en los componentes que conforman las redes podemos encontrar tres niveles:

- **Hardware**: el primer nivel lo forma todo el hardware necesario para establecer las comunicaciones entre los ordenadores. En este nivel hay que destacar que indistintamente del medio de transmisión, ya sea utilizando cables o comunicaciones inalámbricas, y de la tecnología que se utilice (**Ethernet**, Wi-Fi, Bluetooth, etc.) la **tarjeta de red** es el componente hardware principal para poder realizar dichas comunicaciones.

- **Software de red**: el segundo nivel lo compone el conjunto de software de red encargado definir los **protocolos** o ‘normas’ de comunicación entre los computadores. Los protocolos indican como efectuar las comunicaciones lógicas mediante el envío de conjuntos de datos llamados paquetes. Dentro de una red el protocolo **TCP/IP** es el encargado de comunicar los diferentes ordenadores. El uso del protocolo **UDP/IP** también está muy extendido en la actualidad.

- **Software de aplicaciones**: en el último nivel encontramos a todas las aplicaciones encargadas de interactuar con los usuarios de la red y que permiten compartir información y recursos. Dentro de este tipo de software podemos encontrar el modelo **cliente-servidor** donde ordenadores individuales, llamados hosts se conectan a ordenadores que ofrecen servicios, los servidores.

4.2 Tipos de red

Las redes se suelen diseñar en función de las necesidades que suelen ser muy variables ya que dependen de en gran parte de los recursos que se disponga y del propósito para el cual se quiera diseñar la red.

A continuación se muestra un listado de clasificación de tipos de red:

Por alcance:

- Red de área local, o LAN: limitada a un área especial relativamente pequeña tal como un cuarto, un solo edificio, etc. No utilizan medios o redes de interconexión públicos. En la actualidad es la más usada.
- Red de área local inalámbrica, o WLAN: comunicación de datos inalámbrico flexible, alternativa a las redes de área local cableadas.
- Redes de área amplia, o WAN: redes que se extienden sobre un área geográfica extensa utilizando medios como: satélites, cables interoceánicos, Internet, fibras ópticas públicas, etc.
- Otras: Red de área personal, o PAN, Red inalámbrica de área personal, o WPAN (Bluetooth), Red de área de campus, o CAN, Red de área metropolitana o MAN, Red de área de almacenamiento o SAN, Red de área local virtual, o VLAN (Virtual LAN), etc.

Por tipo de conexión:

- Medios guiados (red cableada): cable coaxial, cable de par trenzado, fibra óptica
- Medios no guiados (red inalámbrica): red por radio, red por infrarrojos, red por microondas.

Por topología:

- En bus: se caracteriza por tener un único canal de comunicaciones (denominado bus, troncal o backbone) al cual se conectan los diferentes dispositivos.
- En anillo: cada estación está conectada a la siguiente y la última está conectada a la primera.
- En estrella: las estaciones están conectadas directamente a un punto central y todas las comunicaciones se han de hacer necesariamente a través de éste.
- En malla: cada nodo está conectado a todos los otros.
- En árbol: los nodos están colocados en forma de árbol. Desde una visión topológica, la conexión en árbol es parecida a una serie de redes en estrella interconectadas salvo en que no tiene un nodo central.

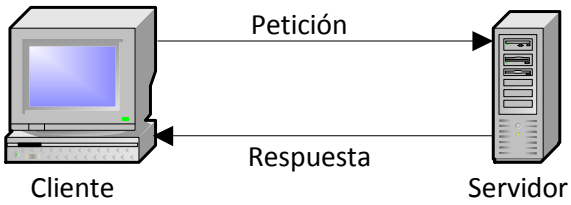
Otras:

- Por grado de autentificación: red privada, red de acceso público.
- Por relación funcional: Cliente-servidor, Peer-to-peer.
- Por tecnología: red Point-To-Point, red broadcast.
- Por la direccionalidad de los datos: Simplex o unidireccional, Half-duplex, Full-duplex, o dúplex
- Por grado de difusión: intranet, Internet.
- Por servicio o función: red comercial, red educativa, etc.

Como se puede observar son muchos los tipos y clasificaciones de red existentes en la actualidad. Para poder escoger que características debería poseer nuestra red antes de montarla se tiene elaborar un diseño de red que nos ayuda a diferenciar los diferentes puntos dentro de red y facilita su implementación.

4.3 Modelo Cliente – Servidor

El modelo cliente servidor es el conjunto de aplicaciones encargadas de distribuir las tareas de ofrecer recursos o servicios, los servidores, y de demandar, los clientes.
La idea es la siguiente:



En este modelo, el cliente es el encargado de empezar la comunicación enviado una petición solicitando determinado servicio a un servidor que es el encargado de proveer el servicio. Esta separación entre cliente y servidor es una separación de tipo lógico donde el servidor es una aplicación que se activa y espera la solicitud de los clientes.

Normalmente clientes y servidores suelen atribuirse a máquinas diferentes y separadas físicamente pero de hecho pueden ser dos áreas separadas en un mismo ordenador. Una misma máquina puede ser cliente y servidor al mismo tiempo dependiendo de la tarea y rol que desempeñe en cada momento. Esto quiere decir que un servidor no tiene porque ejecutarse necesariamente sobre una sola máquina.

Algunas de las ventajas que ofrece este modelo son las siguientes:

- Independencia de hardware y software que se utilice en los clientes / servidores.
- Control centralizado de accesos, recursos y integridad de los datos son controlados por el servidor. Esto facilita la tarea de actualización de recursos o datos e impide que clientes no autorizados puedan acceder a los servicios.
- Escalabilidad horizontal (añadir clientes) y vertical (ampliar potencia de los servidores). Cualquier elemento puede ser aumentado o mejorado, se pueden añadir nuevos nodos a la red (clientes y servidores).
- Integridad de los datos y programas centralizados en servidores que facilitan su integridad y mantenimiento. Es posible reemplazar, reparar, actualizar servidores y sus clientes no se verán afectados.
- Encapsulación de los servicios, los detalles de la implementación de un servició son transparentes al cliente.

- Muchos clientes utilizan los mismos servidores y mediante ellos comparten recursos lógicos y físicos.
- Transparencia de localización física, el cliente no tiene porque saber donde se encuentra situado el recurso que quiere utilizar.

Después de observar que este modelo dispone de un gran número de aspectos positivos vamos a detallar algunas de las desventajas que presenta.

- Saturación del servicio y/o del tráfico de la red. Cuando una gran cantidad de clientes envían peticiones simultáneas al mismo servidor pueden llegar a causar la saturación de éste congestionando el tráfico de red y llegando a impedir que pueda ofrecer el servicio.
- Una caída del servidor provoca que las peticiones no puedan ser atendidas y por lo tanto no pueden ser satisfechas.
- El cliente no dispone de los mismos recursos que pueda existir en el servidor, si el servicio intenta acceder a un recurso del cliente no existente puede provocar errores.
- Se necesita un hardware y software especifico para los servidores ya que de otra forma quizá no se puedan atender todas las peticiones. Esto provoca que el coste aumente.

En el montaje del clúster utilizaremos este modelo donde los servidores ofrecerán servicio a clientes que se podrán conectar a través de internet.

4.4 Tecnología Ethernet

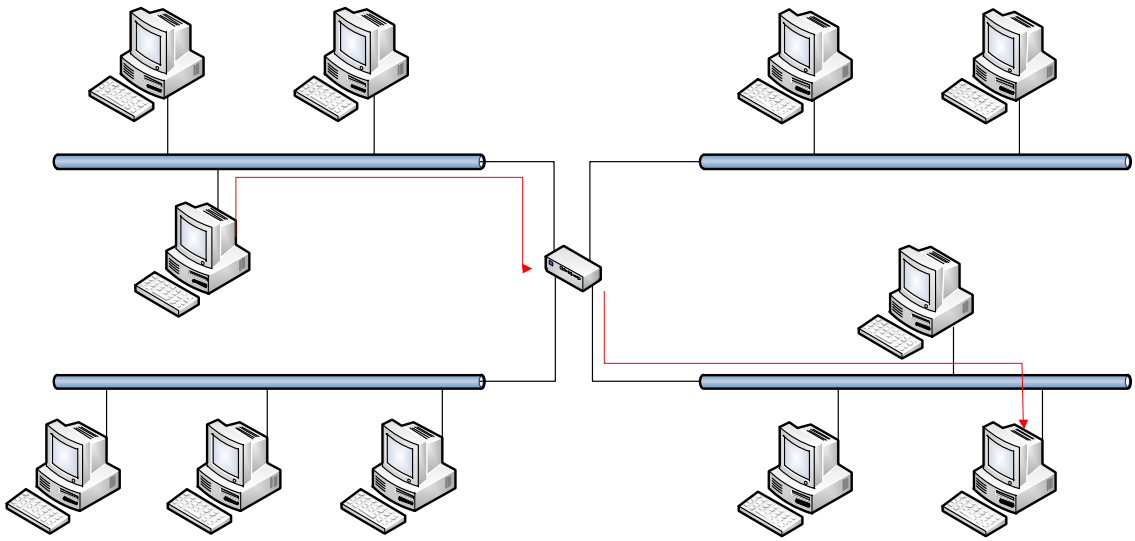
Ethernet es una tecnología de redes de área local basada en la transmisión de tramas. Utiliza el protocolo CSMA/CD (Acceso Múltiple por Detección de Portadora con Detección de Colisiones) en el cual los dispositivos de red escuchan el medio antes de transmitir información. Se aseguran de que el canal está disponible antes de hacer una transmisión.

A lo largo de los años han ido apareciendo y mejorando varias versiones de esta tecnología. En la actualidad podemos hablar de Ethernet como todas las redes locales cableadas y de señalización a nivel físico donde los formatos de trama hacen de enlace entre los diferentes nodos que conforman la red indistintamente del protocolo de acceso. Esto quiere decir que el concepto de esta tecnología se ha extendido sin ser necesario ni obligatorio utilizar el protocolo CSMA/CD.

Los principales elementos que podemos encontrar en una red Ethernet son:

- Tarjetas de red: son los dispositivos que permiten que un ordenador pueda acceder a la red local. Cada tarjeta dispone de un identificador que la distingue del resto llamada MAC.
- Repetidores: elementos que reciben y reenvían las señales. Principalmente aumentan el alcance de la conexión física, se utilizan para asegurarse que las señales no se degraden.

- Conmutadores (Switch): dispositivos que reciben tramas y las transmiten a todos sus puertos exceptuando aquel por donde la recibió guardándose la dirección de respuesta. De esta forma cuando se reciba la contestación el dispositivo pueda enviarla directamente al nodo que inició la comunicación. Principalmente sirve para conectar diferentes nodos y reducir el tráfico de red mejorando funcionamiento.
- Bridge: interconecta segmentos de red haciendo el cambio de tramas entre las redes de acuerdo con una tabla de direcciones que le dice en qué segmento está ubicada una dirección MAC dada. Permite conectar dos redes lan distintas.
- Nodos de red: son los diferentes ordenadores y dispositivos (impresoras, escáneres, etc.) conectados a la red.
- Medio de interconexión: cables mediante a través de los cuales se envía la información (RJ45).



En el diagrama podemos observar como mediante la tecnología Ethernet y los componentes que la forman se pueden conectar varias redes de área local. Esto se extrapola a redes más grandes conectadas a través de internet que utilizan ésta tecnología en sus redes locales.

4.5 Protocolos TCP/IP – UDP/IP

Para que el uso de servicios y accesos a información entre los diferentes ordenadores de una red pueda llevarse a cabo se necesita establecer protocolos que establezcan las bases de la comunicación entre éstos.

Antes de empezar a hablar de los protocolos TCP y UDP vamos a definir el concepto de IP. Una dirección IP es una etiqueta numérica que identifica, de manera lógica y

jerárquica, a un dispositivo. Normalmente las IP's van asociadas a ordenadores aunque otros dispositivos tales como impresoras, faxes, etc. también pueden tener asignada una.

El protocolo IP es un protocolo no orientado a conexión, esto quiere decir que no se establece ningún tipo de conexión previa antes de enviar datos. El flujo de información va en un solo sentido, un nodo envía información a otro sin haber establecido previamente comunicación. Los datos se envían en bloques llamados paquetes.

TCP es un protocolo orientado a conexión. Cuando un nodo envía datos a otro éste confirma la recepción de los mismos. El protocolo garantiza que los datos serán entregados en su destino sin errores. La detección de errores se consigue utilizando el control CRC que mediante una ecuación matemática permite verificar la integridad de los datos transferidos. TCP es uno de los protocolos fundamentales en Internet (junto con el protocolo IP), es utilizado por muchas aplicaciones y conjuntamente con otros protocolos.

Las conexiones TCP se componen de tres etapas:

- Establecimiento de conexión
- Transferencia de datos
- Fin de la conexión

La conexión se establece mediante el procedimiento llamado *3 way handshake*, en el que se configuran algunos parámetros que aseguran la entrega ordenada de los datos. En la desconexión se utiliza el procedimiento de *4-way handshake.*

El protocolo UDP, al igual que sucede con es un protocolo IP, es un protocolo no orientado a conexión, es decir que el flujo de información va en un solo sentido. Esto hace que UDP no garantice la entrega de los paquetes. Algunos paquetes de información pueden perderse sin que el emisor se entere o incluso lleguen desordenados. El ordenador destino no puede confirmar la recepción de la información recibida así como tampoco conocerá al emisor de los datos excepto su IP. Aunque estas características parezcan un inconveniente, el no uso de conexión para enviar los datos hace que sea un protocolo relativamente rápido ya que no introduce retardos en las conexiones.

Por lo tanto podemos decir que TCP está pensado para enviar grandes cantidades de información de forma fiable y esto lo convierte en un en un protocolo más lento que UDP.

4.6 Diseño de Red

Antes de empezar a montar una red se debe realizar un análisis o estudio previo, más conocido como **diseño de red**. Éste análisis consiste en pensar y clasificar las necesidades y los recursos existentes con los que montar dicha red.

Hace falta puntualizar que diseñar correctamente una red es vital para que los recursos sean aprovechados al máximo y no se produzcan pérdidas económicas ni desaprovechamiento de material.

También hay que tener en cuenta que la información debe ser accesible desde todos aquellos puntos que se deseen, la espera de los accesos tiene que minimizarse en la medida de lo posible y los servicios han de poder atender las peticiones eficientemente tanto en tiempo como en capacidad.

A continuación se va a detallar los requisitos a tener en cuenta en el diseño de la red para el clúster de computación.

- Accesibilidad desde internet. La gente ha de ser capaz de enviar peticiones a través de internet y así poder utilizar el servicio del clúster.
- Rápida respuesta del clúster. Intentar servir las peticiones lo antes posible.
- Accesibilidad a zonas de recursos privados. Hay zonas a las que no nos interesa que se tenga acceso desde el exterior.
- Consulta a BBDD. El clúster necesita consultar a base de datos para poder procesar las peticiones que le llegan.
- Seguridad. Poder filtrar el acceso a los recursos.
- División de la red en subredes donde alojar las máquinas encargadas de procesar los procesos del clúster.

Los componentes que necesitaremos para cubrir las necesidades que nos plantea el clúster son los siguientes:

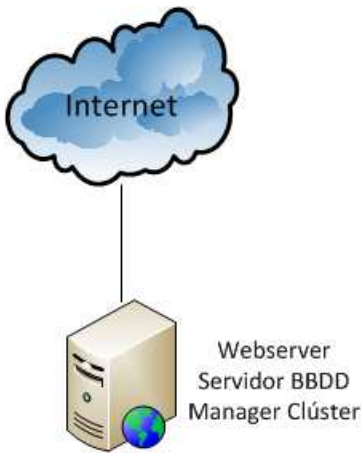
- Un servidor web: Debemos alojar la web en un servidor que atienda las peticiones del exterior y a su vez pueda consultar la base de datos alojada en otro servidor de la red. Debido a su potencial de peligrosidad ya que se está dando acceso desde el exterior el servidor web será alojado en una zona desmilitarizada (DMZ). Así nos aseguraremos que el acceso sea exclusivo a este recurso. Se trata de una zona separada del resto de la red donde solo se puede tener acceso a la red local pasando a través de un firewall que hace de filtro en las comunicaciones.
- Un servidor de BBDD: Se necesita montar un servidor donde alojar la base de datos a la que consultará el clúster.
- Firewall: Imprescindible para garantizar la seguridad de los accesos a los recursos y separar los diferentes activos de la red de posibles ataques de terceros. Debe ser el filtro entre internet y nuestra red y procesar todas las peticiones entrantes y salientes.
- Switch: Encargado de conectar las diferentes máquinas y subredes.

No hay que caer en el error de complicar el diseño de la red. La red ha de ser lo más simple posible siempre y cuando cumpla con todas las necesidades para la cual se requiere. Por eso no se deben poner componentes extras siempre que no sean imprescindibles o mejoren el funcionamiento y rendimiento de la red.

Dado que hay muchas posibles implementaciones que satisfacen el mismo problema a continuación se va a explicar que diseño se adapta más a nuestras necesidades mediante un par de ejemplos que sin llegar a cumplir todos los requisitos serían soluciones validas para el funcionamiento del clúster. Después se indicará cual ha sido el diseño escogido y cuáles son sus ventajas.

Ejemplo 1:

Un posible diseño que cumple las necesidades de ofrecer el servicio del clúster es el que se puede ver en la imagen.



Podemos observar que en muchas ocasiones una red puede ser tan sencilla como sencillo sea el problema que queremos resolver. En nuestro caso con una sola máquina bastaría para cubrir todos los servicios que debe ofrecer el clúster. Es el diseño más simple que podríamos implementar.
Aunque su sencillez es innegable y posiblemente es la primera solución que nos viene a la cabeza cuando pensamos en servicios alojados en internet (páginas web, webservers, etc.) nos podemos llegar a encontrar con los siguientes inconvenientes:

Rendimiento: nuestra máquina no solo debe atender las peticiones que le lleguen a través de la web sino que también debe controlar los accesos a base de datos y a su vez realizar los cálculos pertinentes para poder devolver los resultados en el mínimo tiempo posible. En este caso muy posiblemente el clúster funcionaria por debajo de un rendimiento optimo o deseable.

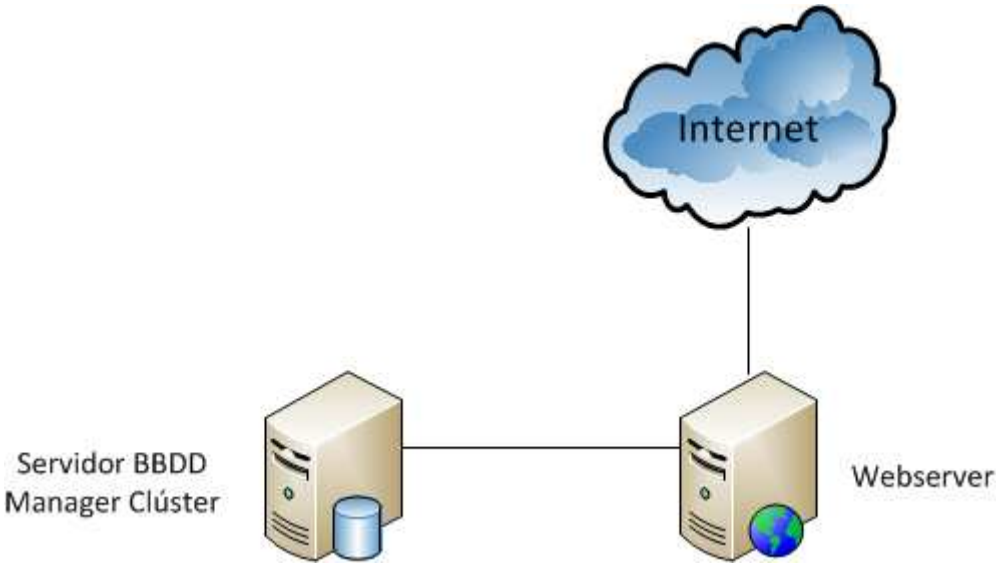
Escalabilidad: se entiende por escalabilidad como la capacidad de cambiar el tamaño o configuración para adaptarse a unas circunstancias cambiantes. Esto quiere decir que en cuanto queramos agregar nuevos servicios o funcionalidades nos encontramos algún que otro problema para hacerlo al tener todo el sistema centralizado en un solo equipo.

Modularidad: capacidad de dividir un sistema en varias partes que interactúan entre sí y que son fácilmente substituibles sin la necesidad de realizar cambios en el resto de partes. Si nuestro servidor falla perderíamos el 100% del servicio y tendríamos que reinstalar todo el sistema.

Seguridad: posiblemente el inconveniente más importante sea este. En este diseño no tan solo comprometemos la web, el servicio con acceso a internet, sino que también la base de datos y la propia máquina. Entendemos por comprometer por el conjunto de acciones que ponen en peligro o en situación de ser vulnerables a ciertas acciones que afectan a la disponibilidad, confidencialidad o integridad de la máquina y todos los datos que ésta contenga. Podríamos decir que el ordenador al quedar expuesto es vulnerable a ataques con intención entre otras cosas de conseguir información confidencial, obtener el control de la propia máquina, etc.

Los principales inconvenientes que presenta este esquema son el del rendimiento y seguridad.

Ejemplo 2:



Este esquema ya es más parecido a la solución que debemos definir. En él ya se han separado algunos servicios mejorando así el rendimiento y otros aspectos como modularidad.

Vamos a analizarlo un poco más en detalle:

Rendimiento: En el nuevo diseño, se han dividido algunos servicios. La carga de trabajo ya no recae en un solo equipo y podemos obtener tiempos de respuesta menores. Los equipos no irán tan saturados.

Escalabilidad: Aunque sigue siendo un esquema sencillo se podrían agregar nuevos servicios a alguno de los servidores. Esto haría que el rendimiento bajase de nuevo

llegando a tener problemas parecidos al ejemplo 1 por lo que podemos decir que presenta un nivel bajo de escalabilidad.

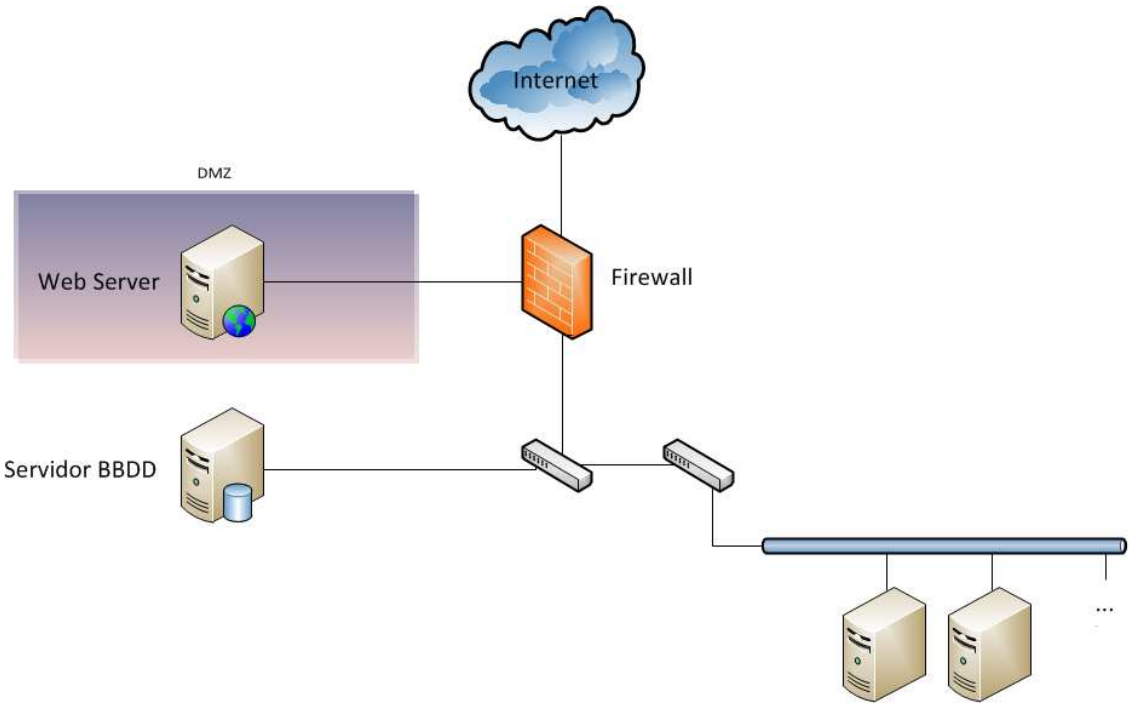
Modularidad: Sin que sea una revolución es sin duda otro aspecto mejorado. Si uno de los dos servidores falla no hace falta reinstalar el sistema por completo.

Seguridad: Sigue sin presentar el nivel de seguridad que requerimos para una red de donde es imprescindible proteger los recursos y la información que contengamos en las base de datos. Sin el nivel de seguridad óptimo se pueden producir pérdidas económicas y de material.

Aunque presenta ciertas ventajas sigue sin ser capaz de resolver el problema de la seguridad de los equipos y de la información. Este aspecto es imprescindible siempre que tratemos con datos sensibles referentes a usuarios o de empresas clientes.

Solución propuesta:

Una vez vistos los dos ejemplos simples, la solución que se propone es la siguiente:



En esta solución se han corregido los errores de seguridad mencionados en los ejemplos anteriores.
Ahora disponemos de una zona *segura* (DMZ) por detrás del firewall. Se han separado los servicios que tienen acceso a Internet de los que no necesitan esta conexión elevando así el nivel de seguridad.

También se han incrementado positivamente el resto de puntos. Con este diseño es bastante fácil agregar elementos a la red. Se pueden incrementar los servicios añadiendo tantas máquinas como necesitemos.
También se mejora el rendimiento, ya que cada servidor está dedicado a un solo servicio.
El fallo de uno de los servidores es fácilmente detectable y si llegase a producirse la situación únicamente deberíamos sustituir el servidor estropeado.

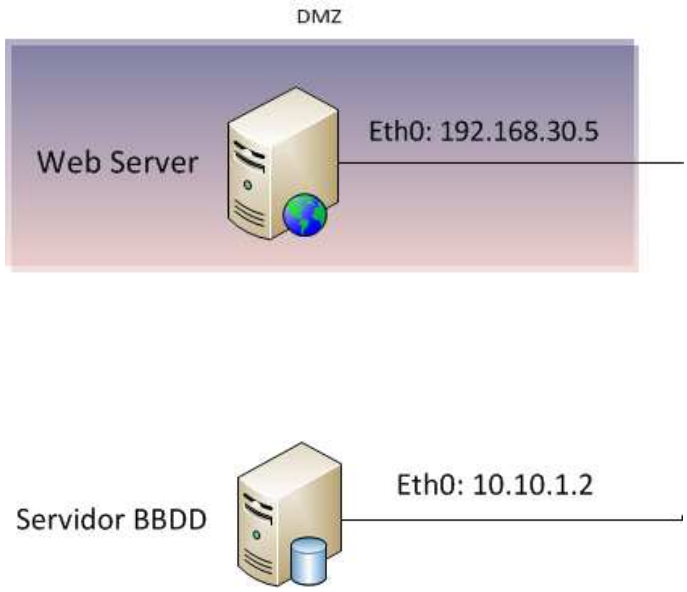
Este diseño cumple todas las características que necesitamos para garantizar tanto el buen funcionamiento del clúster como la seguridad de la red. A partir de aquí se podrían diseñar otros modelos más complejos en los que por ejemplo mejorar algún servicio.

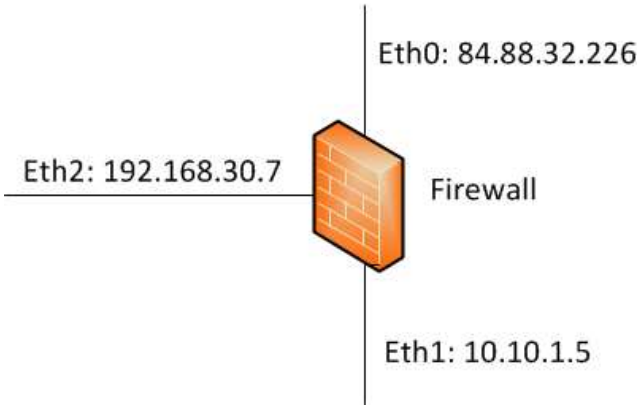
Por último hay que destacar que otro aspecto que debemos tener en cuenta a la hora de diseñar una red es el del material que disponemos para poder montarla. Muchas veces nos encontramos en la situación de querer diseñar sistemas más seguros, fiables o eficientes pero no podemos llevarlo a cabo ya que no disponemos de los componentes y materiales necesarios.

4.7 Especificación de IPs y puertos usados en la instalación

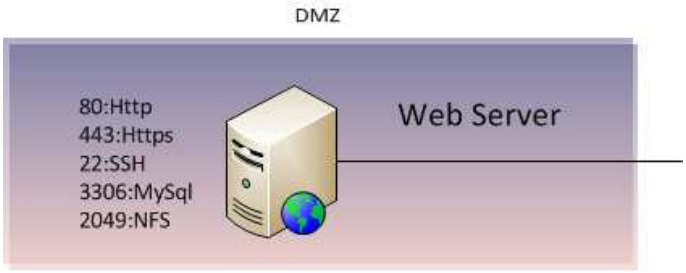
En este apartado se van a detallar que IP se ha asignado a cada máquina así como los controladores Ethernet que se han configurado. También se mostrarán los puertos que utiliza cada servidor para los diferentes servicios.

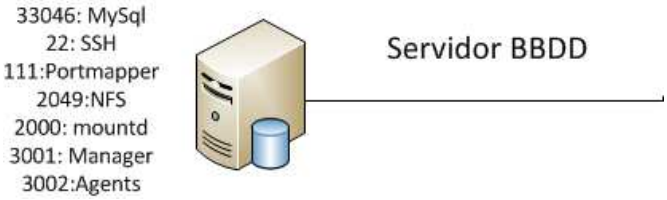
Servidores por Ethernet/IP

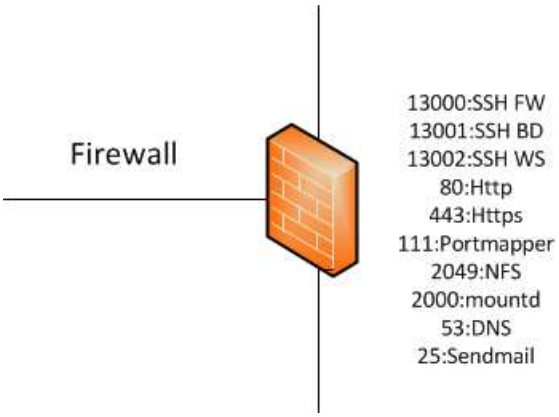




Servidores por Puerto/Servicio







5. Hardware, sistema operativo y aplicaciones

Este proyecto se ha montado totalmente a partir de software gratuito o de licencia libre. Está planteado principalmente para que el coste invertido en software sea inexistente y este aspecto condiciona a la hora de poder elegir que aplicaciones podemos instalar. Aún sin disponer de software de pago hay que destacar que en la actualidad la calidad de las aplicaciones de licencia libre está en un nivel alto donde no tienen nada que envidiar al software privado.

También hay que tener en cuenta que las aplicaciones instaladas muchas veces van sujetas a las restricciones de hardware de las que disponemos. En nuestro caso disponemos de 3 máquinas antiguas que trabajan bajo arquitecturas de 32 bits.

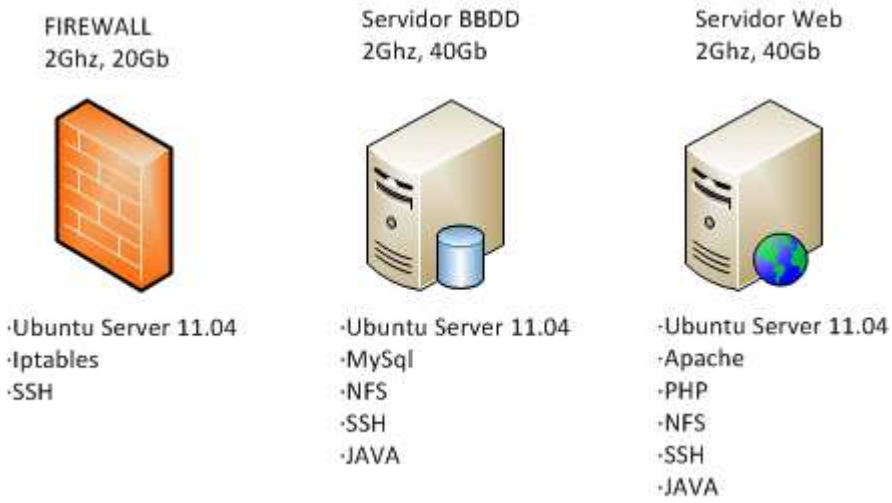
Las especificaciones de las máquinas son las siguientes:

	Procesador	Disco	RAM
Máquina 1:	Intel 2Ghz	40 Gb	256 Mb
Máquina 2:	Intel 2Ghz	40 Gb	256 Mb
Máquina 3:	Intel 2Ghz	20 Gb	256 Mb

Se puede observar que no disponemos de máquinas demasiado potentes y que el espacio en discos del que disponemos es limitado.

A continuación se va a detallar que software se ha instalado y cuales son los aspectos más relevantes en cuanto a sus características y configuraciones. Todos los pasos para su instalación y configuración están detallados en el capitulo 7 (Manual de instalación).

El siguiente esquema muestra que es lo que se ha instalado y sobre que máquinas



El sistema operativo escogido para la instalación de las diferentes máquinas es Ubuntu Server 11.04.

La elección de un sistema operativo que se adapte a nuestras necesidades es imprescindible para no encontrar posteriores problemas de incompatibilidades una vez tengamos todo el sistema montado. Su rendimiento debe ser acorde al propósito para el que se utilice.

Las aplicaciones que aparecen en el gráfico anterior son las que se van a detallar en los próximos puntos de éste capitulo.

A parte de éstas aplicaciones se han instalado un gran número de aplicaciones que aparecen detalladas en el manual de instalación del apartado 7.

Una vez aclarados todos los puntos vamos a ver en detalle el conjunto principal de aplicaciones.

5.1 Ubuntu server

Qué es Ubuntu server:

Ubuntu server es un sistema operativo que utiliza un núcleo Linux basado en Debian. Fácil de instalar y de configurar viene distribuido por una licencia de código abierto y está mantenido por la compañía Canonical.

Dónde surgió:

El primer lanzamiento de Ubuntu fue en 2004 y surgió como una bifurcación del código del SO Debian con la intención de ofrecer una distribución más sencilla y fácil de usar para los usuarios finales. Actualmente su uso está en crecimiento ya que cada vez son más los usuarios que instalan ésta distribución.

Características:

Ubuntu soporta arquitecturas de 32 y 64 bits y viene instalado con un gran número de aplicaciones que facilitan las tareas de mantenimiento y actualización del sistema operativo.
La versión de servidor no trae instalado entorno gráfico.
Se caracteriza principalmente por su compatibilidad con la mayoría de arquitecturas y su fácil instalación y mantenimiento. En las últimas versiones presenta un mejor rendimiento.

Alternativas:

Existe una amplia variedad de alternativas a Ubuntu. Debian o Centos son sistemas operativos de características similares pero que quizá estén un nivel por encima en cuanto a versiones de servidor se refiere. Debian es posiblemente uno de los sistemas más punteros y estables de los existentes en la actualidad.

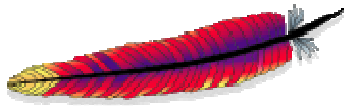
Configuración

Las principales configuraciones realizadas sobre Ubuntu son las de creación de usuarios que acceden al sistema, instalación de aplicaciones, instalación de servicios y configuración de la red.

Para el montaje del clúster todos los servidores se han configurado inicialmente con la configuración por defecto. Se han realizado 3 particiones, 2 particiones EXT4 para raíz (/) y la home y una lógica para la partición SWAP.

Uno de los pasos importantes es el de la configuración de red que nos dará acceso a Internet y la posibilidad de instalar las aplicaciones necesarias para el funcionamiento del sistema.

Todas las configuraciones e instalaciones están detalladas en el capitulo 7 donde se explica paso a paso como realizarlas.



5.2 Servidor HTTP Apache

Qué es apache:

Apache es el servidor Web HTTP de código abierto más utilizado hoy en día. Es desarrollado y mantenido por una comunidad de desarrolladores de Apache Software Foundation.

Donde surgió:

En 1995 comenzó su desarrollo basándose en la aplicación NCSA HTTPd. Era un servidor web desarrollado en el National Center for Supercomputing Applications y que fue cancelado en 1998. Con el paso de los años el código ha sido reescrito hasta ser substituido por lo que conocemos hoy en día.

Características:

Se caracteriza por ser el servidor que presenta más robustez y estabilidad en la actualidad. También destaca por su gran configurabilidad y permitir ejecutarse en múltiples sistemas operativos como Windows, sistemas basados en Unix, Novell, Mac OS y otros.

No posee interfaz gráfica aunque su configuración es bastante sencilla y al ser tan popular es muy fácil encontrar información sobre como realizar dichas configuraciones.

Otra gran característica de este servidor es que es modular. Consta de un core y diversos módulos que permiten ampliar fácilmente las capacidades del servidor. Existe una gran lista de módulos que podemos instalar en el servidor apache o incluso, al ser de código abierto, si se poseen los conocimientos necesarios se pueden programar módulos nuevos. De entre todos los módulos existentes hay que destacar los módulos para el uso de SSL, PHP, PERL y Rewrite.

También posee bases de datos de autenticación y negociado de contenido y permite personalizar la respuesta ante posibles errores que se puedan dar en el servidor. Se puede configurar para que ejecute scripts cuando ocurra un error en concreto.
Permite con mucha facilidad la creación y gestión de logs para tener un mayor control sobre lo que sucede en el servidor.

Alternativas:

Como alternativa al servidor Apache podemos encontrar IIS (Internet Information Services). IIS es un servidor web desarrollado por Microsoft. Como principal diferencia a Apache el IIS presenta interfaz gráfica. Se caracteriza también por ser fácil de configurar y de ser bastante estable. En la actualidad viene preinstalado en las versiones más recientes de windows.

Otra alternativa es NGINXES. Se trata de un servidor HTTP desarrollado para una de las webs más visitadas de Rusia. Posee buena estabilidad aunque está aún en versión beta. Consume pocos recursos y soporta SSL.

Configuración:

El archivo principal de configuración de Apache es **apache2.conf** aunque dependiendo del sistema operativo donde se instale podemos encontrarlo con el nombre de httpd.conf.

Para la instalación del clúster se ha habilitado el modulo de rewrite que permite la reescritura de direcciones. Se utiliza normalmente para simular páginas estáticas pero que realmente son dinámicas.

Los diferentes módulos de los que dispone apache los podemos encontrar en la carpeta /etc/apache2/mods-available. Cuando activamos un modulo también se crea un softlink en la carpeta /etc/apache2/mods-enabled.

También se ha modificado el archivo charset que se encuentra en la carpeta /etc/apache2/config.d/ para que apache permita el uso de acentos en las páginas.

Se ha creado un nuevo archivo en la carpeta /etc/apache2/sites-available/ llamado cluster. En este archivo un virtualhost para la página web del clúster.

```
 GNU nano 2.2.6                                          Archivo: cluster

<VirtualHost *:80>
        ServerAdmin webmaster@localhost
        ServerName cluster.com
        ServerAlias www.cluster.com

        DocumentRoot /var/www/cluster

        <Directory /var/www/cluster>
                AllowOverride All
                Allow from All
        </Directory>

        CustomLog ${APACHE_LOG_DIR}/cluster.com-access.log combined
</VirtualHost>
```

Es necesario crear un softlink en la carpeta /etc/apache2/sites-available/ que apunte al archivo creado en /etc/apache2/sites-available/

Cada vez que realicemos un cambio tendremos que reiniciar el servicio llamado apache2.

5.3 Mysql

Qué es MySql:

MySql es un gestor de base de datos relacional multiusuario y multihilo. Es desarrollado y mantenido como software libre por la empresa MySql AB.

Donde surgió:

En 1995 surge la idea de crear MySql a la empresa sueca MySql AB con el principal objetivo de cumplir el estándar SQL. MySql está escrito en C y C++ y en la actualidad el desarrollo está a cargo de Sun Microsystems quien posee los derechos sobre MySql AB.

Características:

Destaca por su adaptación a muchos entornos de desarrollo, permitiendo interactuar con lenguajes de programación como php, java o perl. También destaca su integración en distintos sistemas operativos.

Es una plataforma robusta que aguanta hasta 50 millones de registros y ofrece un sistema de contraseñas y privilegios seguro. Utiliza la verificación basada en el host y el tráfico de contraseñas está cifrado en los accesos al servidor.

También es un sistema multihilo que permite la ejecución de varias tareas simultáneamente y multiusuario que permite definir diferentes roles con los que poder realizar diferentes tareas sobre las bases de datos.

Permite escoger entre múltiples motores de almacenamiento para cada tabla y agrupa transacciones de varias conexiones para incrementar el número de transacciones por segundo. Estas dos últimas características son especificas y solo se encuentran en Mysql.

En la actualidad es uno de los SGBD más utilizados a nivel mundial. Su fácil manejo e instalación junto al gran abanico de posibilidades que nos ofrece lo convierten en una opción puntera a la hora de montar nuestras bases de datos.

Alternativas:

PostgreSQL es una buena alternativa al uso de MySql ya que presenta algunas características muy parecidas en cuanto a rendimiento. De licencia gratuita permite escrituras y accesos simultáneos a tablas sin necesidad de bloqueos. Multiplataforma destaca principalmente por su robustez.

Oracle considerado como uno de los SGDB más completos que destaca por su estabilidad, escalabilidad y soporte de transacciones.
Es multiplataforma, permite el uso de particiones para la mejora de eficiencia y replicación. Es un sistema de base de datos con mucha orientación a aplicaciones internet. El inconveniente más destacable es su precio.

Configuración:

El archivo de configuración de mysql se encuentra en la carpeta /etc/mysl y se llama my.cnf

En este archivo encontramos las configuración principal de mysql. Con el propósito de realizar las diferentes pruebas que se han ejecutado durante la instalación del clúster (rendimiento de mysql) se han ido haciendo modificaciones en este archivo. En la siguiente imagen se muestra como se ha variado el número de threads de 100 a 1000 y el número de conexiones de 100 a 10000.

```
max_allowed_packet      = 16M
thread_stack            = 192K
thread_cache_size       = 8
# This replaces the startup script and checks MyISAM tables if needed
# the first time they are touched
myisam-recover         = BACKUP
max_connections        = 10000
#table_cache            = 64
thread_concurrency     = 1000
#
# * Query Cache Configuration
#
query_cache_limit       = 1M
query_cache_size        = 16M
#
# * Logging and Replication
#
# Both location gets rotated by the cronjob.
# Be aware that this log type is a performance killer.
# As of 5.1 you can enable the log at runtime!
#general_log_file        = /var/log/mysql/mysql.log
#general_log             = 1
```

Posteriormente hemos creado un usuario para que se pueda conectar a la base de datos del clúster sin poner en riesgo el resto de BD.

```
Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.
This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL v2 license

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> GRANT ALL PRIVILEGES ON cluster.* to web@192.168.30.5 IDENTIFIED BY "xxxxxx"
```

Utilizando la instrucción GRANT se crea un usuario que tiene acceso a la base de datos cluster.

Aunque mysql nos ofrece una consola donde poder ejecutar todas las sentencias SQL es muy recomendable instalar las aplicaciones de administración, siempre que se disponga de entorno gráfico, que nos facilitan enormemente la tarea de gestión de MySql.

5.4 Iptables

Qué es Iptables:

Iptables es una herramienta que permite filtrar paquetes red y realizar traducciones de direcciones de red (NAT: mecanismo utilizado por enrutadores IP para intercambiar paquetes entre dos redes que poseen IPs incompatibles).

Donde surgió:

Iptables forma parte de un framework llamado Netfilter que permite interceptar y manipular paquetes de red. El proyecto comenzó en 1998 a cargo de Rusty Russell quien al año siguiente fundó Netfilter Core Team. Se trata del grupo de personas que llevan el desarrollo y mantenimiento del proyecto.

Funcionamiento:

Iptables mediante una serie de reglas permite definir que hacer con los paquetes de red que circulan por una red. Las reglas se agrupan en cadenas y éstas a su vez en tablas que están asociadas a un tipo en concreto de procesamiento.

Junto con cada regla se define un objetivo. Si iptables detecta que algún paquete cumple alguna de las reglas definidas éste realizará la acción que haya sido definida como objetivo. El análisis de las reglas es secuencial esto quiere decir que cada paquete que pase por nuestro ordenador será comparado con cada regla que se haya definido en iptables salvo que cumpla alguna de ellas. En el caso de cumplirse alguna regla la secuencia se detiene y el objetivo de la regla indica que se ha de hacer con el paquete. Por el contrario si no coincide con ninguna de las reglas establecidas la política por defecto establece que hacer con el paquete.

Estructura:

Iptables presenta la siguiente estructura de tablas y reglas:

- Filter table: Tabla de filtrado, donde se bloquean permite que un paquete continúe su camino. Todos los paquetes pasan a través de ésta tabla.
Contiene las siguientes cadenas:

- **Input**: Todos los paquetes destinados al propio ordenador donde se encuentran las reglas definidas.
- **Output**: Todos los paquetes que deben ser enviados fuera del ordenador
- **Forward**: Todos los paquetes que han de ser encaminados a su destino dentro de nuestra red.

- **Nat table**: Tabla de traducción de direcciones de red, es la responsable de reescribir la dirección o el puerto a donde debe ir destinado un paquete.
Contiene las siguientes cadenas:

- **Prerouting**: Paquetes entrantes a nuestra red y que no van destinados a la propia máquina.

- **Postrouting**: Paquetes salientes de nuestra red y que necesitan reescribir su dirección para que lleguen a su destino.
- **Output:** Limitado a paquetes generados localmente

- Mangle table: Tabla responsable de ajustar las opciones de los paquetes. Modifican los paquetes.
Contiene todas las cadenas aparecidas anteriormente:

- **Prerouting**
- **Input**
- **Forward**
- **Output**
- **Postrouting**

Configuración:

Iptables se configura escribiendo cada una de sus reglas mediante el comando **iptables.** Si queremos reconfigurar las reglas para iptables deberemos hacer un limpiado de las caches donde están reglas previamente cargadas.

Dado que iptables no contiene ningún archivo donde podamos guardar las reglas es aconsejable escribirlas en un script que se ejecute al arrancar la máquina o bien utilizar la directiva que guarda las reglas y las pone por defecto. Esta segunda opción tiene como inconveniente que si queremos cambiar alguna deberemos volver a escribir todas y cada una de las reglas que necesitemos (a no ser que las tengamos en guardadas en algún archivo).

El script que se ha realizado para la configuración del clúster es el siguiente:

```
#!/bin/sh
##SCRIPT IPTABLES
echo -n Reconfigurando Reglas iptables

##Variables
Ip_EXT="84.88.32.226"
Ip_BD="10.10.1.2"
Ip_WS="192.168.30.5"
eth_EXT="eth0"
eth_BD="eth1"
eth_WS="eth2"



##FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
```

```
#Politica por defecto ALL DENY
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

#Bit de Forward
echo 1 > /proc/sys/net/ipv4/ip_forward

#Localhost total acceso
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

#Servicio web
iptables -t nat -A PREROUTING -i $eth_EXT -p tcp --dport 80 -j DNAT --to
$Ip_WS:80
iptables -A FORWARD -d $Ip_WS -p tcp --dport 80 -j ACCEPT

#ssh firewall
iptables -A INPUT -p tcp --dport 13000 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 13000 -j ACCEPT

#ssh bd
iptables -t nat -A PREROUTING -i $eth_EXT -p tcp --dport 13001 -j DNAT --to
$Ip_BD:22
iptables -A FORWARD -d $Ip_BD -p tcp --dport 22 -j ACCEPT

#ssh bd
iptables -t nat -A PREROUTING -i $eth_EXT -p tcp --dport 13002 -j DNAT --to
$Ip_WS:22
iptables -A FORWARD -d $Ip_WS -p tcp --dport 22 -j ACCEPT

#Navegacion desde firewall a webs(http) y webs seguras(https)
iptables -A INPUT -p tcp -m tcp --sport 80 -m state --state RELATED,ESTABLISHED
-j ACCEPT
iptables -A OUTPUT -p tcp -m tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp -m tcp --sport 443 -m state --state
RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -p tcp -m tcp --dport 443 -j ACCEPT
#Config DNS
iptables -A INPUT -i $eth_EXT -p tcp -m tcp --sport 53 -j ACCEPT
iptables -A OUTPUT -o $eth_EXT -p tcp -m tcp --dport 53 -j ACCEPT



iptables -A INPUT -i $eth_EXT -p udp -m udp --sport 53 -j ACCEPT
iptables -A OUTPUT -o $eth_EXT -p udp -m udp --dport 53 -j ACCEPT

#Acceso NFS
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p tcp --dport 2049 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p tcp --sport 2049 -j
```

```
ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p udp --dport 2049 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p udp --sport 2049 -j
ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p tcp --dport 111 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p tcp --sport 111 -j
ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p udp --dport 111 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p udp --sport 111 -j
ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p tcp --dport 2000 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p tcp --sport 2000 -j
ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p udp --dport 2000 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p udp --sport 2000 -j
ACCEPT

#Acceso de la DMZ a la BD
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p tcp --dport 3306 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p tcp --sport 3306 -j
ACCEPT
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

#Permitir hacer ping a host remotos
iptables -A INPUT -i $eth_WS -p ICMP -j ACCEPT
iptables -A OUTPUT -o $eth_WS -p ICMP -j ACCEPT
iptables -A INPUT -i $eth_BD -p ICMP -j ACCEPT
iptables -A OUTPUT -o $eth_BD -p ICMP -j ACCEPT
```

Hay que destacar la complejidad del uso de Iptables. Aún siendo una herramienta muy potente muchas veces es difícil de configurar. Es muy fácil confundir reglas o poner algunas que hacen lo contrario a lo que deseamos. No dispone de entorno gráfico ni debug. Los logs son de poca ayuda ya que solo se escriben si se cumple una regla determinada lo que dificulta la tarea de seguimiento.

5.5 SSH

Qué es SSH:

SSH (Secure Shell) es un protocolo que sirve para conectarse remotamente a las máquinas de una red y que permite, mediante el interprete de comandos, controlar por completo éstas.

Donde surgió:

SSH surgió como mejora del protocolo telnet. La diferencia principal es que SSH utiliza técnicas de cifrado y la información viaja encriptada. La primera versión fue escrita por Tatu Ylönen en 1995. A principios de 1999 se empezó a escribir la versión que conocemos actualmente implementada por OpenBSD y más conocida como OpenSSH.

Características:

La principal característica de SSH es la encriptación de sus comunicaciones. Todos los datos enviados y recibidos durante una sesión se transfieren por medio de una encriptación de 128 bits. Esto hace que descifrar la comunicación sea una tarea extremadamente difícil.

SSH también proporciona un medio seguro para el uso de aplicaciones gráficas siempre que se disponga de un servidor que las soporte.

Una de las ventajas de que este protocolo encripte las comunicaciones es que se puede usar para asegurar otros protocolos inseguros. Se puede convertir en un conducto para transformar en seguros otros protocolos inseguros mediante el uso de la técnica de reenvío por puerto.

Utilizando este protocolo en los inicios de sesión de Shell remota y para copiar archivos entre máquinas se reduce las amenazas de seguridad. El cliente SSH y el servidor usan firmas digitales para verificar su identidad. Cada paquete está cifrado por una clave conocida solo por el sistema local y el remoto.

Alternativas:

La alternativa a SSH es precisamente el protocolo al que intenta mejorar. Telnet sirve para controlar máquinas remotamente pero sin el cifrado de las comunicaciones. Solo permite el uso de la consola.
Varios usuarios se pueden conectar a una máquina, abrir sesión, siempre que tengan acceso para hacerlo, y trabajar utilizando ésta misma máquina.

El principal inconveniente de éste protocolo es el que presenta en temas de seguridad. Es fácil interceptar las comunicaciones y al no ser cifradas obtener toda la información que circula entre las máquinas que han establecido conexión.

Configuración:

El archivo de configuración de ssh se llama sshd_config y se encuentra en la carpeta /etc/ssh/

Se ha modificado el puerto de conexión de ssh para que las conexiones al firewall se hagan por el puerto 13000 en vez por el puerto por defecto 22. También se cambiado la línea que deja que un usuario se pueda logear como root.

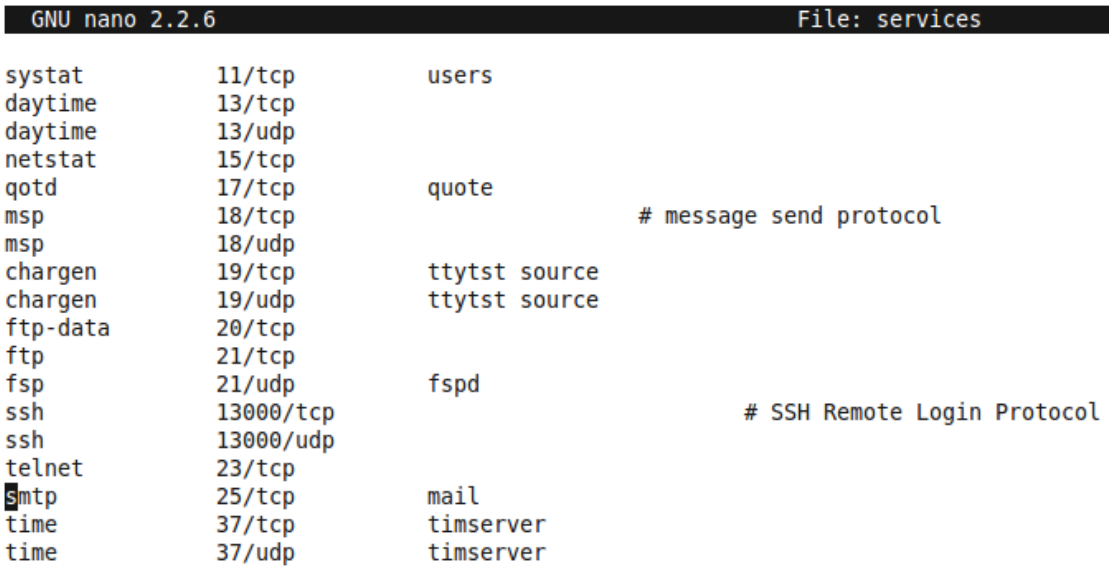
Port 13000
PermitRootLogin no

```
  GNU nano 2.2.6                                          File: sshd_config

# Package generated configuration file
# See the sshd_config(5) manpage for details

# What ports, IPs and protocols we listen for
Port 13000
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::
#ListenAddress 0.0.0.0
Protocol 2
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 768

# Logging
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes
```

Las firmas digitales se guardan en la misma carpeta que el archivo de configuración. Son los tres archivos que se pueden observar en la imagen. ssh_host_rsa_key, ssh_host_dsa_key y ssh_host_ecdsa_key.

El otro archivo que se ha modificado es services. Se encuentra en la carpeta /etc/. Se han modificado las líneas que definen el puerto de escucha del protocolo ssh

ssh 13000/tcp
ssh 13000/udp



Una vez se realizadas las modificaciones en los archivos de configuración se tiene que reiniciar el servicio llamado ssh.

5.6 NFS

Qué es NFS:

NFS (Network File System) es un protocolo utilizado para compartir archivos entre las máquinas de una red. Permite a hosts remotos montar archivos sobre la red e interactuar con éstos como si estuviesen montados localmente.

Donde surgió:

Fue desarrollado en 1984 por Suns Microsystems con el objetivo de poder compartir archivos independientemente de la máquina, sistema operativo o protocolo de transporte.

Características:

El protocolo NFS utiliza el modelo cliente-servidor. Los clientes acceden de forma remota al recurso compartido en el servidor. Esto representa una ventaja en cuanto a espacio en disco de los clientes ya que no necesitan tener los archivos en local. Los datos se encuentran centralizados y pueden ser modificados por varios usuarios.

También permite compartir dispositivos tales como CD-Rom o memorias zip.

En la actualidad hay 3 versiones de NFS.
NFSv2: ampliamente soportada por muchos sistemas operativos.
NFSv3: en esta versión cabe destacar el uso de manejo de archivos de tamaño variable.
NFSv4: incluye seguridad kerberos (protocolo de autenticación de redes).

Todas las versiones pueden utilizar el protocolo TCP sobre redes IP, de hecho la versión NFSv4 lo requiere. Las otras dos versiones también pueden funcionar mediante UDP.

Las operaciones que realiza este protocolo son síncronas. En caso de una solicitud de escritura, el servidor escribirá físicamente los datos en el disco, y si es necesario, actualizará la estructura de directorios, antes de devolver una respuesta al cliente. Esto garantiza la integridad de los ficheros.

Los servicios que necesarios para que NFS funcione son:

- **Nfs**: Inicia los procesos RPC apropiados para servir peticiones para los sistemas de archivos compartidos NFS.
- **Nfslock**: Un servicio opcional que inicia los procesos RPC adecuados para permitir que clientes NFS bloqueen archivos en el servidor.
- **Portmap**: El servicio RPC para Linux; responde a las peticiones para servicios RPC y configura las conexiones al servicio RPC solicitado. No se utiliza con NFSv4.

El puerto por defecto que utiliza NFS es el 2049.

Alternativas:

La alternativa a NFS puede ser SAMBA. Es una implementación basada en el protocolo que utilizan los archivos compartidos de Windows.
Samba posee la característica de poder compartir archivos entre sistemas GNU/Linux y Windows. Los usuarios de windows ven los recursos como carpetas de red locales y los usuarios de GNU/Linux pueden montar en sus sistemas de archivos estas unidades de red como si fueran dispositivos locales.

Configuración:

El archivo de configuración por defecto de NFS lo encontramos en /etc/default/ y se llama nfs-kernel-server. En este archivo debemos modificar la línea que indica que puerto debe utilizar el servicio rpc cada vez que se conecte el servidor. Si no se modificase el puerto el servicio rpc escogería uno al azar y no podríamos filtrarlo a través del firewall.

Modificamos la línea RPCMOUNTDOPTS=”--port 2000” para fijar el puerto por el que se conectará.

```
  GNU nano 2.2.6                                        File: nfs-kernel-server

# Number of servers to start up
# To disable nfsv4 on the server, specify '--no-nfs-version 4' here
RPCNFSDCOUNT=8

# Runtime priority of server (see nice(1))
RPCNFSDPRIORITY=0

# Options for rpc.mountd.
# If you have a port-based firewall, you might want to set up
# a fixed port here using the --port option. For more information,
# see rpc.mountd(8) or http://wiki.debian.org/?SecuringNFS
#RPCMOUNTDOPTS=--manage-gids
RPCMOUNTDOPTS="--port 2000"

# Do you want to start the svcgssd daemon? It is only required for Kerberos
# exports. Valid alternatives are "yes" and "no"; the default is "no".
NEED_SVCGSSD=

# Options for rpc.svcgssd.
RPCSVCGSSDOPTS=
```

El archivo donde se indican los recursos compartidos se encuentra en la carpeta /etc/ y se llama exports. En este archivo hemos indicado el recurso que queremos compartir con el servidor web. Esto se realiza en el servidor de BBDD porque es el que contiene los archivos.

```
 GNU nano 2.2.6                                          File: /etc/exports

# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#

/home/bd/BD 192.168.30.5/255.255.255.0(rw,sync,no_subtree_check)
```

En este caso se ha compartido la carpeta BD con el host que posee la ip 192.168.30.5

Cada vez que se realiza un cambio en este archivo hay que reiniciar el servicio llamado nfs-kernel-server.

En el cliente nfs tenemos de modificar el archivo fstab que se encuentra en /etc/ para que al iniciar la máquina se monte automáticamente el recurso.

```
 GNU nano 2.2.6                                  Archivo: /etc/fstab

# /etc/fstab: static file system information.
#
# Use 'blkid -o value -s UUID' to print the universally unique identifier
# for a device; this may be used with UUID= as a more robust way to name
# devices that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
proc            /proc           proc    nodev,noexec,nosuid 0       0
# / was on /dev/sda1 during installation
UUID=c959d15b-78ef-4bef-8187-4dca94ee64ff /               ext4    errors=remount-ro 0       1
# swap was on /dev/sda5 during installation
UUID=ce723654-be6d-4621-967c-062ae201f771 none            swap    sw              0       0
/dev/fd0        /media/floppy0  auto    rw,user,noauto,exec,utf8 0       0
#192.168.30.67:/home/bd/BD      /home/webserver/tmp_prueba      nfs     user,exec,dev,nosuid,rw,nfsvers=3,auto  0
10.10.1.2:/home/bd/BD   /home/webserver/tmp_prueba      nfs     user,exec,dev,nosuid,rw,nfsvers=3,auto  0       0
```

El recurso compartido BD se montará en la carpeta tmb_prueba cada vez que se inicie la máquina.

5.7 PHP

Qué es PHP:

PHP es un lenguaje interpretado de alto nivel diseñado principalmente para la creación de páginas web dinámicas. Se trata de un lenguaje de código abierto de uso muy extendido a día de hoy.

Donde surgió:

A finales de 1994 Rasmus Lerdorf diseño la primera versión de PHP. En la actualidad The PHP Group se encarga de la implementación del lenguaje.

Características:

Php se ejecuta principalmente en la parte del servidor, es decir, se ejecuta en el servidor web justo antes de que se envíe la página web al cliente. Esta característica nos proporciona la opción de tener acceso a base de datos.
Presenta gran capacidad de conexión con la mayoría de sistemas gestores de base de datos de lo cuales destaca su conectividad con MySql y PostgreSQL.

Se trata de un lenguaje multiplataforma compatible con la mayoría de SO y con un gran número de servidores web. Lo cual hace que en la actualidad sea uno de los lenguajes más utilizado en la creación de webs dinámicas.

Es un lenguaje de fácil uso basado en perl, c y java que nos permite incrustar trozos de código en mitad del html. El resultado de la ejecución de PHP es código html y dado que normalmente se ejecuta en el servidor la página resultante servida al cliente es únicamente en html, compatible con todos los navegadores web.

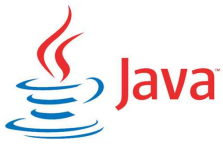
También presenta un amplio manejo de excepciones facilitando el trabajo de los desarrolladores y administradores de las webs que hayan sido programadas con éste lenguaje.

Alternativas:

Una de las alternativas a PHP es ASP.NET . Se trata de un framework para aplicaciones web desarrollado por Microsoft.
Trabaja principalmente con formularios web y se ejecuta en la parte del servidor.
Se suele programar en archivos separados del código html conocidos como code-behind donde el desarrollador escribe códigos referentes a eventos en vez de realizar un recorrido lineal de la página. Está diseñado para programar en entorno Windows.

Podemos encontrar otra alternativa en Coldfusión. Se trata de un servidor de aplicaciones y un lenguaje de programación diseñados para desarrollar aplicaciones web.
Multiplataforma y ampliamente soportado en la mayoría de servidores web.
No es un lenguaje de bases de datos pero usa SQL estándar para recuperar la información de éstas.

5.8 Plataforma Java



Qué es Java:

Java es un entorno que sirve para ejecutar aplicaciones desarrolladas por el lenguaje de programación java o otros lenguajes que compilen a nivel de bytecode (código intermedio más abstracto que el código máquina).

Dónde surgió:

Es creada en 1995 por Suns Microsystems y surge como una idea revolucionaria de la mezcla de muchas tecnologías y que ofrece un entorno de desarrollo y entorno de ejecución en tiempo real. Actualmente pertenece a Oracle Corporation.

Características:

Esta plataforma ofrece un amplio conjunto de herramientas para desarrollo así como la posibilidad de ejecutar aplicaciones. Java no se puede considerar sistema operativo ni hardware especifico ya que funciona por encima de otras plataformas que si trabajan a nivel de hardware (SO). Es más bien considerada una máquina virtual (JVM) que ejecuta aplicaciones utilizando un conjunto de bibliotecas estándar.

Es multiplataforma y está soportada por la mayoría de sistemas operativos de la actualidad. Java se basa en la idea de un procesador virtual que ejecuta programas escritos en el lenguaje de programación Java. No fue la primera plataforma en aparecer que se basase en este concepto aunque si ha sido la que más difusión ha obtenido.

Alternativas:

La plataforma .Net de Microsoft es la principal alternativa. Surge como competencia directa a la plataforma de Java.
A diferencia de la JVM la plataforma .NET es un componente de software que puede ser añadido al sistema operativo Windows y que permite ejecutar programas escritos en ésta misma plataforma.

Las ventajas que ofrecen este tipo de plataformas a la hora de ejecutar aplicaciones no se están quedando en el olvido y son muchas las empresas que se han unido a Microsoft en el desarrollo y mejora de la plataforma .NET.

Configuración:

Se ha instalado la plataforma java principalmente para poder ejecutar las aplicaciones encargadas de procesar la información que llega al clúster (Manager, PendingInfo y Agents).

6. Pruebas

Las pruebas son tareas que se deberían realizar siempre se monta una red para garantizar su fiabilidad, seguridad y disponibilidad.

Es de suma importancia realizar pruebas que nos aseguren que nuestro sistema está protegido contra posibles ataques o accesos no deseados.

También es muy importante realizar pruebas que nos demuestren que nuestro sistema funcionará tal y como se espera.

Las pruebas nos permiten evaluar el nivel de seguridad de nuestra red, nos aportan datos y nos indican el camino a seguir en cuanto a mejora. También nos aportan rendimientos y carencias, nos enseñan como se comportará el sistema si se cumplen ciertas condiciones y nos permiten adelantarnos a posibles contratiempos.

Es cierto que las pruebas no son la solución a todos los problemas que pueda tener una red pero gracias a ellas se pueden descubrir fallos críticos y huecos de seguridad.

Dado que es imposible realizar pruebas sobre todos los servicios y aplicaciones instaladas en las máquinas de una red es vital establecer cuales son los puntos a los que más atención hay que prestar.

En este proyecto se han realizado dos tipos de pruebas:

- Pruebas de rendimiento (stress).
- Pruebas de vulnerabilidades conocidas.

Las pruebas realizadas se han centrado en:

- Rendimiento MySql
- Rendimiento Apache
- Vulnerabilidad Web (Sql Injection)
- Vulnerabilidad Firewall && Apache (Ataque DDOS)

Las pruebas sobre vulnerabilidad realizadas se han centrado en la zona más crítica, la parte que está expuesta a internet (aplicación web y firewall).

6.1 Pruebas stress MySql

Para medir el rendimiento de nuestro MySql se va a realizar un test de Benchmark. La herramienta típica para hacer Benchmark de MySql es sysbench y se puede descargar de la página http://sysbench.sourceforge.net/

Una vez descargada la descomprimimos en una carpeta utilizando el comando tar y posteriormente la instalamos.

```
$ sudo tar -xvf sysbench-0.4.12.tar.gz
$ cd sysbench-0.4.12
$ ./configure
$ make
$ make install
```

Ya tenemos instalada la aplicación ahora debemos configurar el entorno sobre el que hacer las pruebas. Para ello hemos creado una base de datos de prueba a la que llamamos test.

Con la siguiente línea prepararemos la base de datos de prueba:

```
$ sysbench --test=oltp --db-driver=mysql --mysql-db=test --mysql-user=root --mysql-password=root
 --mysql-table-engine=innodb --oltp-table-size=1000000 prepare
```

Y a continuación la tabla sobre la que se efectuaran los accesos:

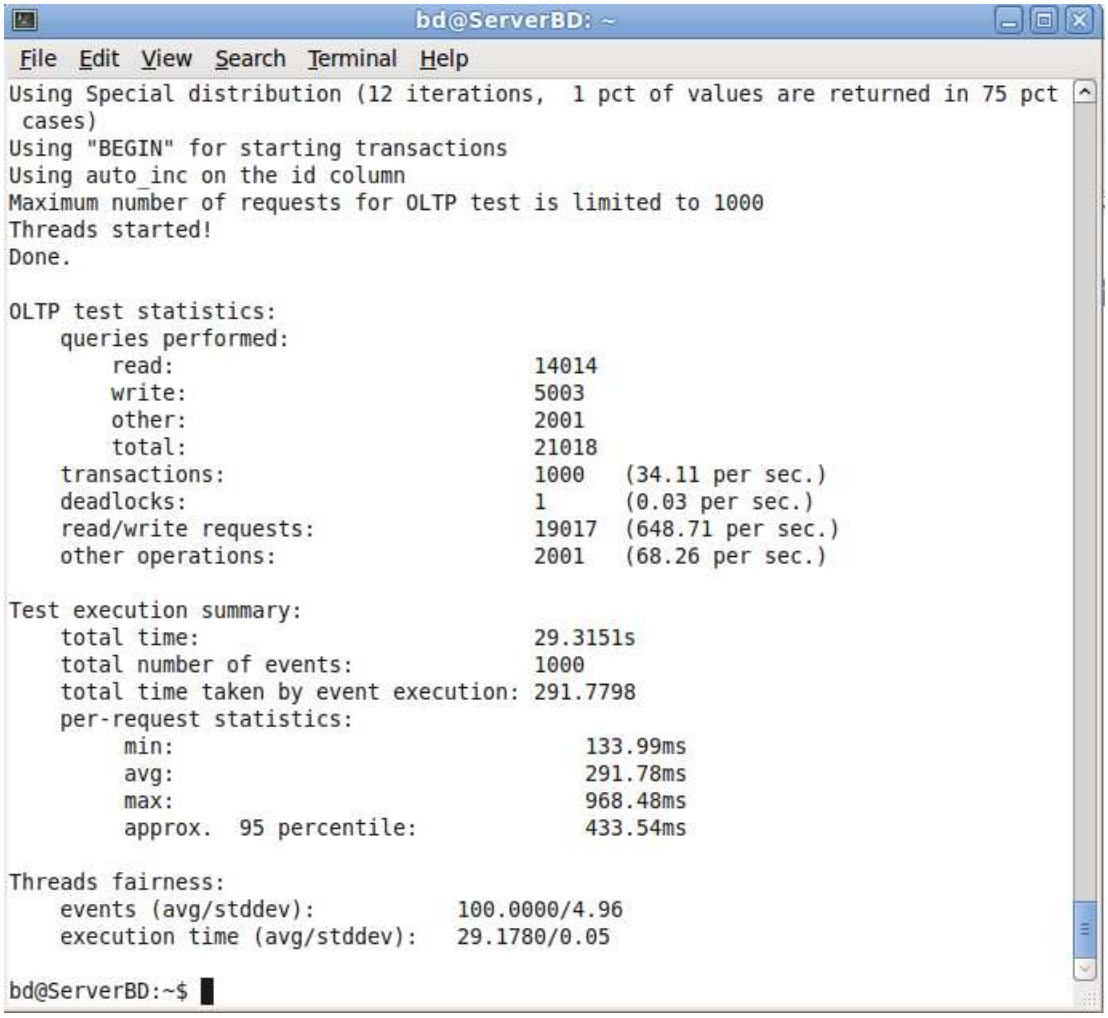
```
Mysql > CREATE TABLE `sbtest` (`id` int(10) unsigned NOT NULL auto_increment,
`k` int(10) unsigned NOT NULL default '0', `c` char(120) NOT NULL default '',
`pad` char(60) NOT NULL default '', PRIMARY KEY  (`id`), KEY `k` (`k`));
```

Ahora ya tenemos el entorno listo para poder empezar a ejecutar el test Benchmark.

El primer test lo vamos a ejecutar con 10 threads(hilos) realizando 1000 peticiones.

```
$ sysbench --test=oltp --db-driver=mysql --mysql-db=test --mysql-user=root --mysql-password=root
--max-requests=1000 --num-threads=10 run
```

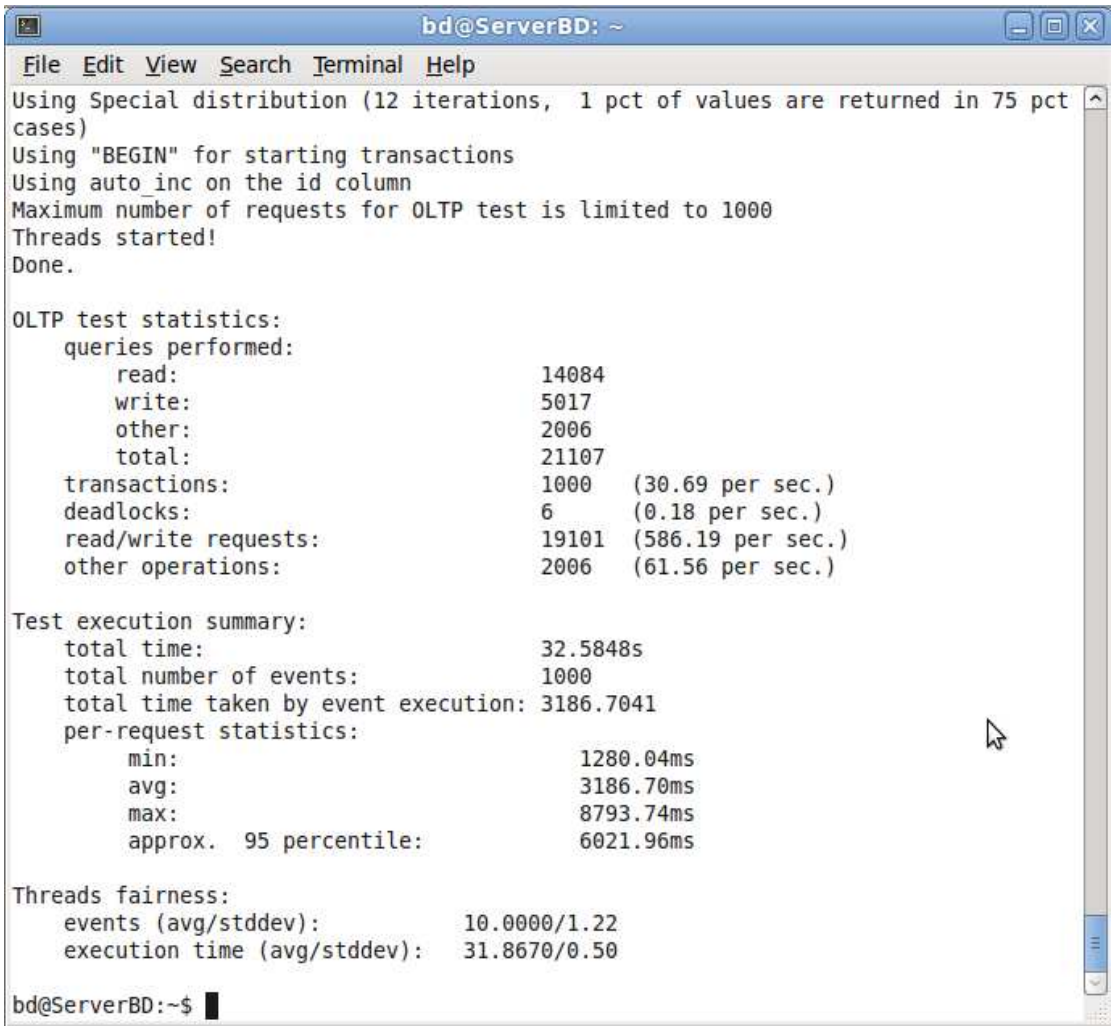
El resultado es el siguiente:



En el segundo test vamos a incrementar el número de threads pasado de 10 a 100 y vamos a mantener el número de peticiones.

$ sysbench --test=oltp --db-driver=mysql --mysql-db=test --mysql-user=root --mysql-password=root
--max-requests=1000 --num-threads=100 run

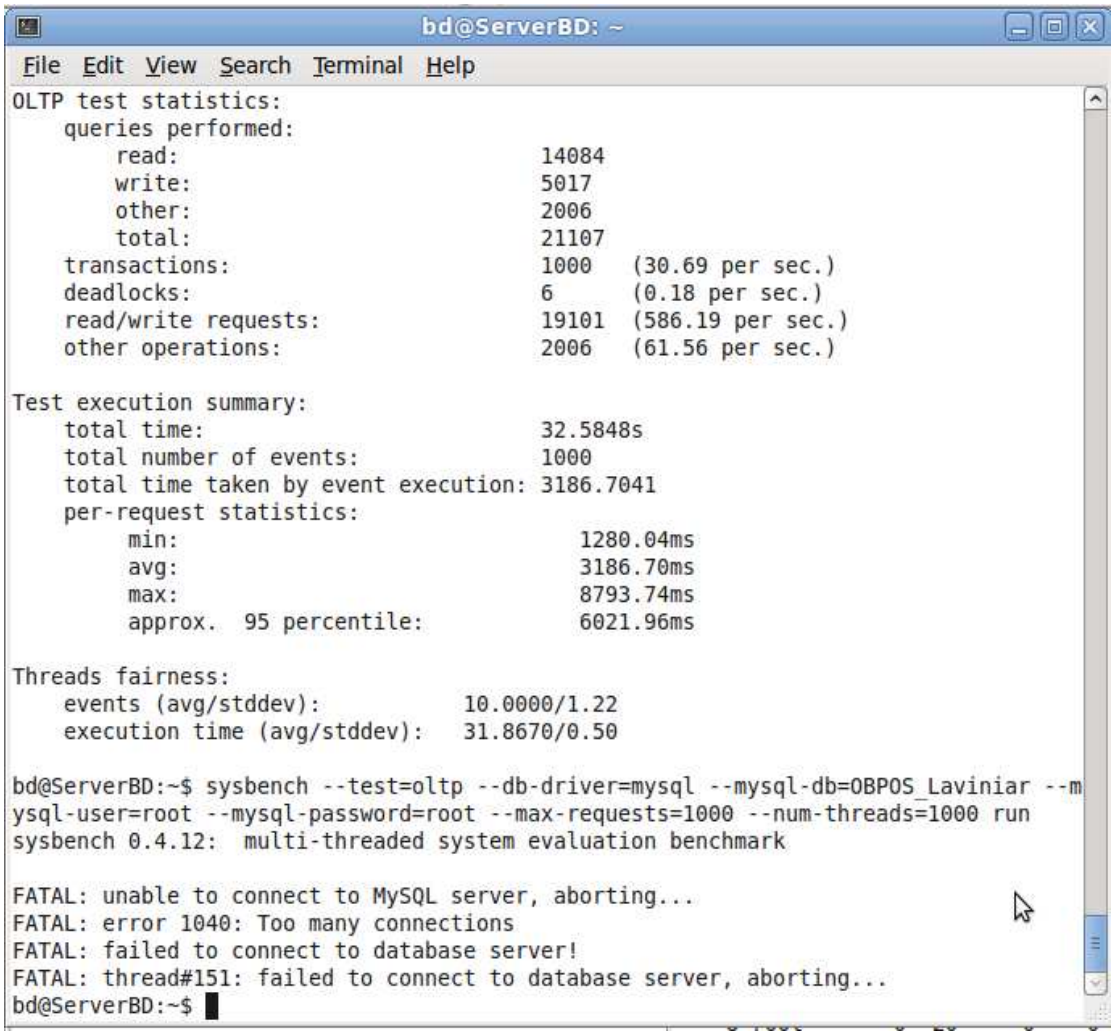
Obteniendo el resultado de:



El resultado ha cambiado considerablemente, todos los tiempos de servicio son mayores que en el test anterior

Ahora vamos a incrementar más aún el número de threads. En este caso vamos a poner 1000 y vamos a dejar que el número de peticiones sigan siendo las mismas.

$ sysbench --test=oltp --db-driver=mysql --mysql-db=test --mysql-user=root --mysql-password=root
--max-requests=1000 --num-threads=1000 run



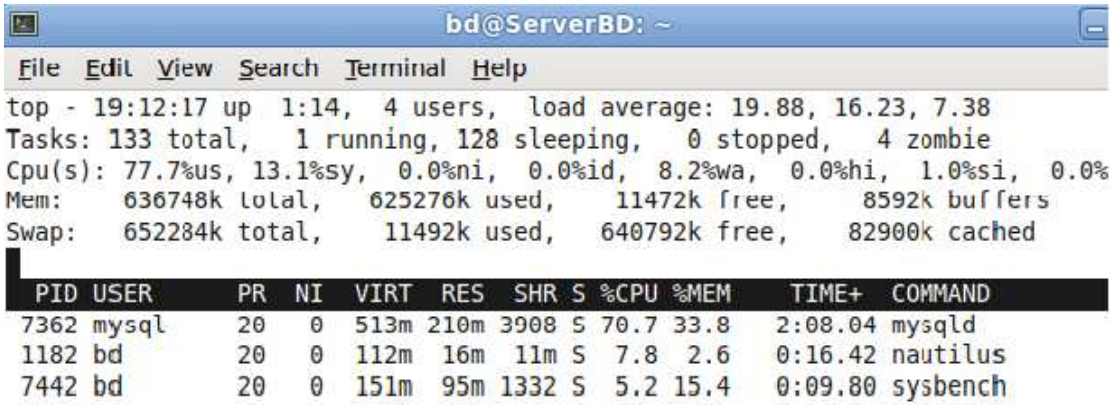
Al realizar este test nos ha devuelto un error que nos indica que hemos superado el límite de conexiones con el que tenemos configurado mysql.
Vamos a cambiar la configuración de mysql para permitir más accesos simultáneos a la base de datos y así repetir el test para observar que sucede.
Para cambiar este parámetro tenemos que abrir el archivo /etc/mysql/my.cnf y modificar la línea donde se indica el máximo de conexiones permitido.

```
myisam-recover         = BACKUP
max_connections        = 10000
#table_cache            = 64
thread_concurrency     = 1000
#
```

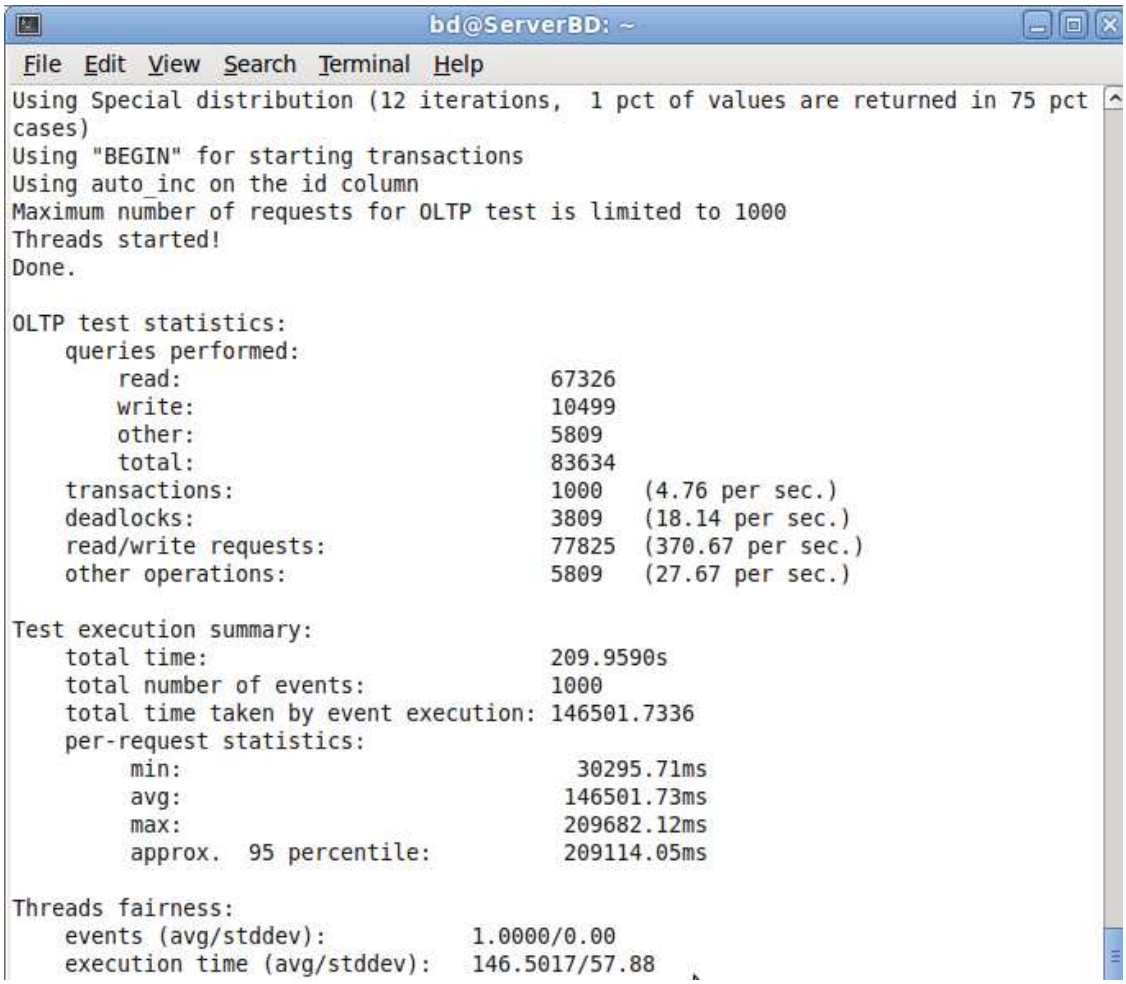
Reiniciamos el servicio de mysql

```
$ Sudo /etc/init.d/mysql restart
```

Y volvemos a ejecutar el test



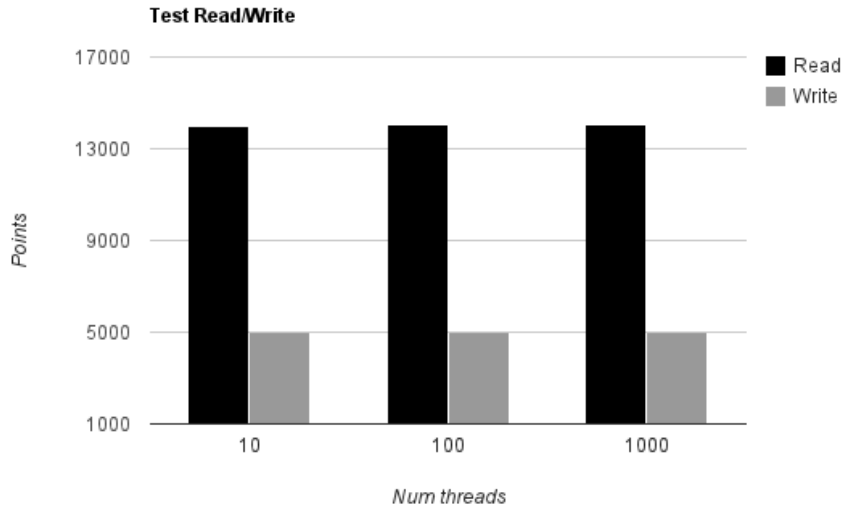
Si hacemos un top de la máquina mientras se ejecuta el test observaremos que la máquina empieza a ir muy justa y el % de cpu ocupado es bastante cercano al 100%. La máquina comienza a ir lenta y el resto de procesos se quedan en espera.



Aquí tenemos el resultado del test. Si lo observamos detenidamente vemos que los tiempos de respuesta han aumentado de forma alarmante. Quizá si llegásemos a tener un nivel de carga tan elevado el servido de base de datos acabaría por saturarse y perderíamos el servicio.

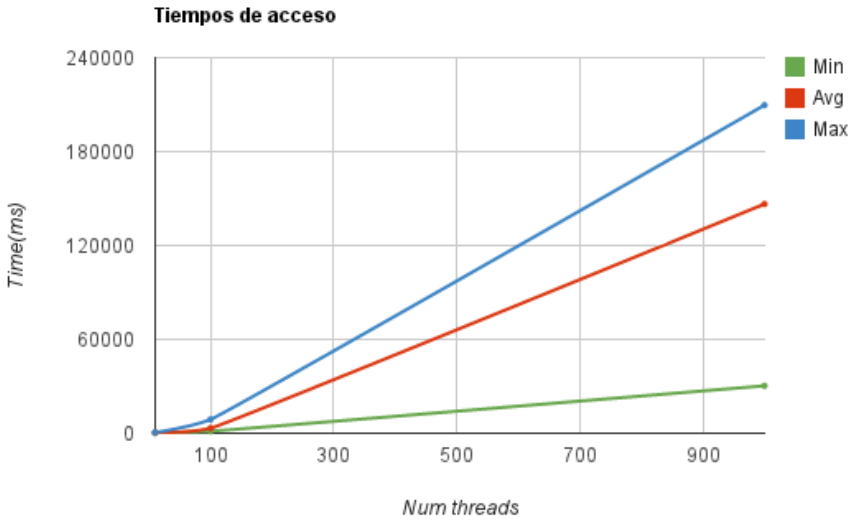
Con los datos obtenidos en los tests anteriores se pueden realizar dos gráficos que analicen lo sucedido durante las pruebas.

La primera gráfica muestra la relación de lecturas y escrituras en función del número de threads.



Contrariamente a lo que podríamos pensar vemos que el rendimiento de MySql se mantiene constante a pesar de aumentar el número de threads. Con esto podemos suponer que el rendimiento no es proporcional al número de máquinas que haya ya que se mantiene a pesar de que se aumenten el número de threads. Si tuviésemos un mayor número de visitas en nuestra web no necesitaríamos aumentar el hardware porque no se necesitarían más máquinas.

La segunda gráfica muestra el tiempo de los accesos en función del número de threads.



Los tiempos de respuesta en cambio si aumentan. A medida que aumentamos los threads los tiempos de acceso aumentan de forma casi exponencial. A mayor número de threads más lentas son las respuestas de las peticiones y esto provoca la ralentización de la aplicación, en nuestro caso la web.

En el caso de que aumentasen las visitas a nuestra web, si quisiésemos mantener un buen tiempo de respuesta de nuestro servidor mysql tendríamos que mejorar el hardware de la máquina por otro más potente. Con nuestro diseño de red esto no es ningún problema ya que se trata de un diseño modular y tan sólo nos veríamos obligados a configurar la parte del servicio de base de datos.

6.2 Pruebas stress Apache

Para medir el rendimiento de apache utilizaremos la herramienta de benchmark que nos proporciona el propio apache conocida con el nombre de **ab.**

ab nos permite simular peticiones a nuestro servidor apache y genera unas estadísticas con las cuales podemos ver el rendimiento que se obtiene. Normalmente se instala con la instalación de apache pero si no lo podemos hacer con el comando

```
$ sudo apt-get install apache2-utils
```

Se ejecuta mediante la siguiente sintaxis:

```
ab [ -A auth-username:password ] [ -c concurrency ] [ -C cookie-name=value ] [
-d ] [ -e csv-file ] [ -g gnuplot-file ] [ -h ] [ -H custom-header ] [ -i ] [ -
k ] [ -n requests ] [ -p POST-file ] [ -P proxy-auth-username:password ] [ -q ]
[ -s ] [ -S ] [ -t timelimit ] [ -T content-type ] [ -v verbosity] [ -V ] [ -w
] [ -x <table>-attributes ] [ -X proxy[:port] ] [ -y <tr>-attributes ] [ -z
<td>-attributes ] [http://]hostname[:port]/path
```

Los parámetros que nos interesa modificar para ver la carga del servidor son la **n** (número de peticiones) y la **c** (número de concurrencias de cada petición).

A continuación se van a mostrar los datos obtenidos por una serie de test realizados en el servidor web. Las pruebas se realizan sobre la misma máquina donde se lanzan los test (localhost).

El primer test lo vamos a realizar con 100 peticiones y 10 concurrencias (threads).

```
$ ab -n100 -c10 http://localhost/
```

```
Document Path:          /
Document Length:        10123 bytes

Concurrency Level:      10
Time taken for tests:   0.321 seconds
Complete requests:      100
Failed requests:        0
Write errors:           0
Total transferred:      1032900 bytes
HTML transferred:       1012300 bytes
Requests per second:    311.71 [#/sec] (mean)
Time per request:       32.081 [ms] (mean)
Time per request:       3.208 [ms] (mean, across all concurrent requests)
Transfer rate:          3144.20 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   0.9      0       3
Processing:     3   31  14.2     27     110
Waiting:        1   11  12.4      6      84
Total:          4   32  14.5     27     110

Percentage of the requests served within a certain time (ms)
  50%     27
  66%     29
  75%     31
  80%     32
  90%     51
  95%     62
  98%     78
  99%    110
 100%    110 (longest request)
```

El primer test se ha realizado con la velocidad que se espera cuando se hacen peticiones a un servidor web. Vemos que los tiempos por petición son bajos lo que indica que el servidor trabaja en perfectas condiciones.

En el siguiente test vamos a aumentar el número de concurrencias por petición de 10 a 100.

$ ab -n100 -c100 http://localhost/

```
Document Path:          /
Document Length:        10123 bytes

Concurrency Level:      100
Time taken for tests:   0.323 seconds
Complete requests:      100
Failed requests:        0
Write errors:           0
Total transferred:      1032900 bytes
HTML transferred:       1012300 bytes
Requests per second:    309.67 [#/sec] (mean)
Time per request:       322.929 [ms] (mean)
Time per request:       3.229 [ms] (mean, across all concurrent requests)
Transfer rate:          3123.57 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        7   14   4.5     14      24
Processing:    21  188  75.6    194     287
Waiting:       21  167  77.6    173     286
Total:         36  202  75.9    207     311

Percentage of the requests served within a certain time (ms)
  50%    207
  66%    247
  75%    266
  80%    282
  90%    304
  95%    309
  98%    311
  99%    311
 100%    311 (longest request)
```

Se ha realizado el test con relativa velocidad aunque ya podemos ver que el tiempo de respuesta por petición ha aumentado en aproximadamente 300 ms. También se puede ver que a medida que se van sirviendo las peticiones el tiempo de estas va creciendo.

El siguiente test lo realizaremos con 1000 peticiones y disminuiremos el número de concurrencias a 10

$ ab -n1000 -c10 http://localhost/

```
Document Path:          /
Document Length:        10123 bytes

Concurrency Level:      10
Time taken for tests:   3.362 seconds
Complete requests:      1000
Failed requests:        0
Write errors:           0
Total transferred:      10329000 bytes
HTML transferred:       10123000 bytes
Requests per second:    297.42 [#/sec] (mean)
Time per request:       33.622 [ms] (mean)
Time per request:       3.362 [ms] (mean, across all concurrent requests)
Transfer rate:          3000.07 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   0.3      0       3
Processing:     3   33  12.0     29     116
Waiting:        1    7   8.9      2      69
Total:          4   33  12.0     29     116

Percentage of the requests served within a certain time (ms)
  50%     29
  66%     32
  75%     36
  80%     41
  90%     52
  95%     58
  98%     66
  99%     75
 100%    116 (longest request)
```

Se puede ver que ha bajado el tiempo dedicado a cada petición y los tiempos de conexión se asemejan a los obtenidos en el primer test ya que si nos fijamos en los mínimos y los máximos son casi idénticos.

Vamos a continuar observando el rendimiento de nuestro servidor ahora aumentando de nuevo el número de concurrencias.

$ ab -n1000 -c100 http://localhost/

```
Document Path:          /
Document Length:        10123 bytes

Concurrency Level:      100
Time taken for tests:   3.348 seconds
Complete requests:      1000
Failed requests:        0
Write errors:           0
Total transferred:      10329000 bytes
HTML transferred:       10123000 bytes
Requests per second:    298.72 [#/sec] (mean)
Time per request:       334.759 [ms] (mean)
Time per request:       3.348 [ms] (mean, across all concurrent requests)
Transfer rate:          3013.19 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    2   5.1      0      29
Processing:    21  320  50.9    329     447
Waiting:       21  286  49.0    296     421
Total:         36  322  47.4    329     447

Percentage of the requests served within a certain time (ms)
  50%    329
  66%    333
  75%    338
  80%    342
  90%    351
  95%    357
  98%    366
  99%    414
 100%    447 (longest request)
```

Volvemos a obtener resultados parecidos a los del segundo test en cuanto a tiempo dedicado por petición pero claramente si observamos la tabla de tiempos de conexión se ve un aumente significativo.

Podemos ya a empezar a ver que el rendimiento del servidor es peor cuando aumentan las concurrencias de las peticiones existentes. Si aumentamos el número de peticiones y disminuimos el número de concurrencias los tiempos mejoran, mientras que si aumentamos el número de concurrencias por cada petición los tiempos de servicio aumentan de forma considerable.

En el último test hemos ya hemos visto que el servidor no ha realizado el test con la facilidad con la que hacia el primero. Para poner al límite el rendimiento se va a ejecutar un último test donde se aumentará por 10 el número de concurrencias por petición.

$ ab -n1000 -c1000 http://localhost/

```
webserver@Webserver:~$ ab -n1000 -c1000 http://localhost/
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking localhost (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
Completed 700 requests
Completed 800 requests
apr_socket_recv: Connection reset by peer (104)
Total of 850 requests completed
```

El test no se ha podido terminar ya que el server no ha soportado el número de peticiones que se le había indicado al ab. El server ha desconectado el socket al haber sido desbordado y sólo se han completado 850 peticiones.

Ahora se van a mostrar los datos obtenidos de los tests realizados desde el exterior, fuera de nuestra red.

$ ab -n100 -c10 http://84.88.32.226/

```
Document Path:          /
Document Length:        10123 bytes

Concurrency Level:      10
Time taken for tests:   16.570 seconds
Complete requests:      100
Failed requests:        0
Write errors:           0
Total transferred:      1032900 bytes
HTML transferred:       1012300 bytes
Requests per second:    6.04 [#/sec] (mean)
Time per request:       1656.952 [ms] (mean)
Time per request:       165.695 [ms] (mean, across all concurrent reques
ts)
Transfer rate:          60.88 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:       91  116  33.3    109     280
Processing:   838 1510 318.9   1592    2143
Waiting:      114  505 108.0    507     752
Total:        933 1626 313.7   1703    2254

Percentage of the requests served within a certain time (ms)
  50%   1703
  66%   1820
  75%   1877
  80%   1891
  90%   1985
  95%   2141
  98%   2225
  99%   2254
 100%   2254 (longest request)
```

Como se intuía los tiempos han subido en una gran proporción. Mientras que antes se atendían unas 311 peticiones por segundo en la última prueba hemos pasado a atender 6 peticiones por segundo. Esto demuestra que el ancho de banda influye en el tiempo de respuesta.

$ ab -n100 -c100 http://84.88.32.226/

```
Document Path:          /
Document Length:        10123 bytes

Concurrency Level:      100
Time taken for tests:   19.273 seconds
Complete requests:      100
Failed requests:        0
Write errors:           0
Total transferred:      1032900 bytes
HTML transferred:       1012300 bytes
Requests per second:    5.19 [#/sec] (mean)
Time per request:       19273.400 [ms] (mean)
Time per request:       192.734 [ms] (mean, across all concurrent reques
ts)
Transfer rate:          52.34 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:       93  193  57.7    194     292
Processing:   872 11059 5340.4  12921   19132
Waiting:      220 6348 3392.2   6353   11415
Total:        973 11252 5379.0  13113   19272

Percentage of the requests served within a certain time (ms)
  50%  13113
  66%  14799
  75%  15568
  80%  15763
  90%  17128
  95%  18259
  98%  19244
  99%  19272
 100%  19272 (longest request)
```

$ ab -n1000 -c10 http://84.88.32.226/

```
Document Path:          /
Document Length:        10123 bytes

Concurrency Level:      10
Time taken for tests:   162.510 seconds
Complete requests:      1000
Failed requests:        0
Write errors:           0
Total transferred:      10329000 bytes
HTML transferred:       10123000 bytes
Requests per second:    6.15 [#/sec] (mean)
Time per request:       1625.098 [ms] (mean)
Time per request:       162.510 [ms] (mean, across all concurrent reques
ts)
Transfer rate:          62.07 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:       89  123  66.6    111     676
Processing:   825 1499 121.9   1493    2215
Waiting:      122  506  92.9    490    1646
Total:        921 1622 128.8   1604    2377

Percentage of the requests served within a certain time (ms)
  50%   1604
  66%   1629
  75%   1666
  80%   1670
  90%   1736
  95%   1832
  98%   1993
  99%   2091
 100%   2377 (longest request)
```

$ ab -n1000 -c100 http://84.88.32.226/

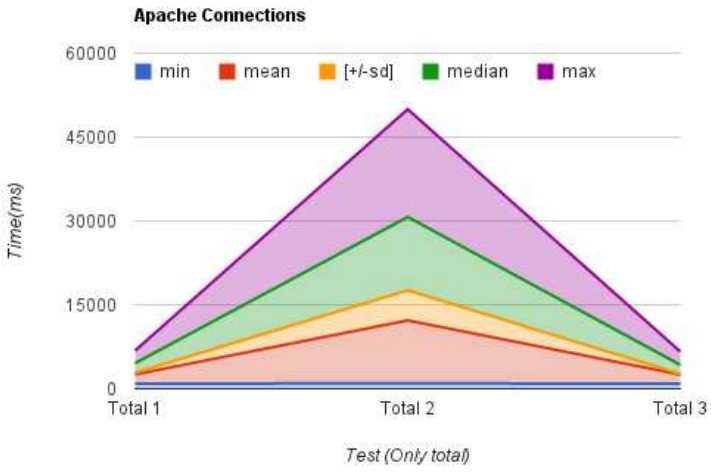
```
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 88.12.41.206 (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
Completed 700 requests
Completed 800 requests
Completed 900 requests
apr_poll: The timeout specified has expired (70007)
Total of 991 requests completed
```

Hemos obtenido error antes de llegar al máximo de ocurrencias que habíamos probado en localhost. Mientras que en localhost el sistema dio fallo en el test de 1000 x 1000 desde el exterior se ha producido mucho antes viéndose reducida la cantidad de peticiones atendidas.

Con los datos de las pruebas desde el exterior obtenemos la siguiente tabla donde se indican los tiempos de cada acción realizada durante las conexiones.

	Min	Mean	[+/-sd]	Median	Max
Connect:	91	116	33.3	109	280
Processing	838	1519	318.9	1592	2143
Waiting	114	505	108	507	752
Total 1	933	1626	313.7	1703	2254
Connect:	93	193	57.7	194	292
Processing	872	11059	5340.4	12921	19132
Waiting	220	6348	3392.2	6353	11415
Total 2	973	11252	5379	13113	19272
Connect:	89	123	66.6	111	676
Processing	825	1499	121.9	1493	2215
Waiting	122	506	92.9	490	1646
Total 3	921	1622	128.8	1604	2377

ab -n100 -c10 http://84.88.32.226/ ab -n100 -c100 http://84.88.32.226/ ab -n1000 -c10 http://84.88.32.226/



De las pruebas realizadas sobre el servidor Apache podemos deducir que:

- Hay un cambio de rendimiento dependiendo de la cantidad de threads.
- El ancho de banda influye en el rendimiento de respuestas y en el rendimiento general de apache que aparte de ralentizar el sistema puede llegar a colapsar el servicio.
- Posiblemente debamos cambiar el hardware del servidor web por uno más potente si el número de conexiones a nuestra página es considerable. Hay que remarcar que el servidor tan sólo posee tan sólo 256Mb de RAM y seguramente se tendrá que aumentar en un futuro.

6.3 SQL Injection

La inyección de SQL consiste en explotar una vulnerabilidad existente en el código de una aplicación web. Se envía sentencias sql camufladas en las transacciones que realiza nuestra aplicación web con la base de datos.

Si los atacantes consiguen ejecutar estás líneas de código malicioso pueden poner en peligro los datos que tengamos dentro de la base de datos.

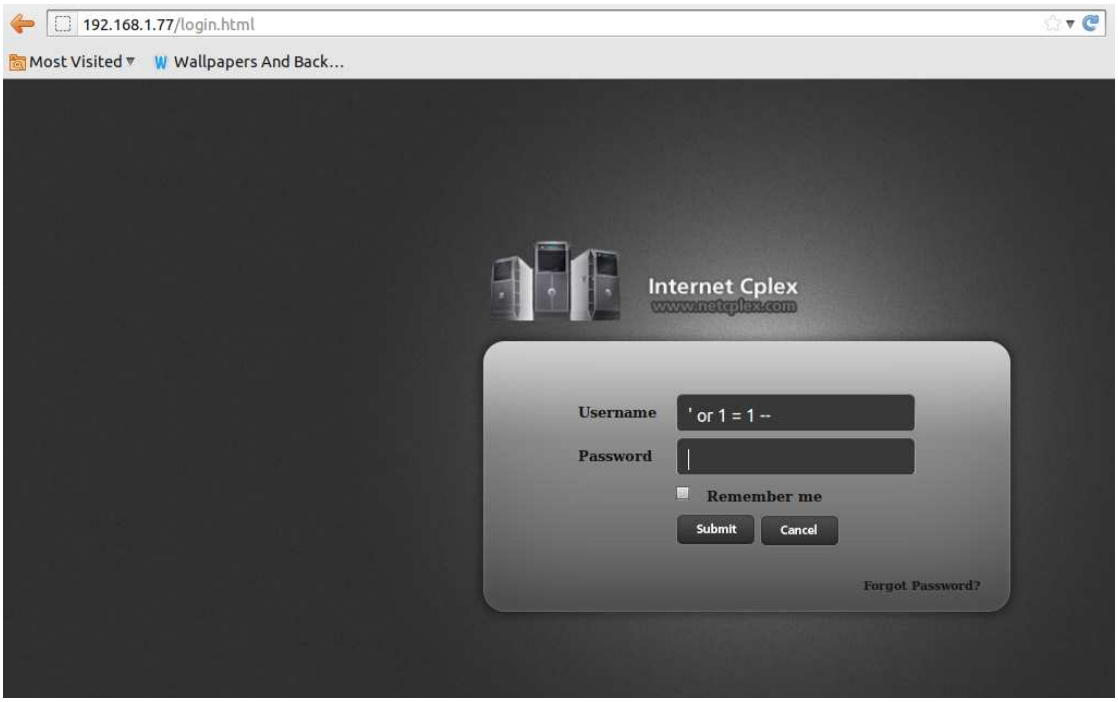
Este tipo de ataques son bastante comunes en la actualidad y suelen producirse sobre aplicaciones web. Realmente la vulnerabilidad viene dada por una deficiente programación. Si el desarrollador no es cuidadoso a la hora de escribir su código impidiendo así que intrusos puedan inyectar a nuestro sistema trozos de sentencias SQL nuestra base de datos corre un grave peligro.

La idea es aprovechar esa programación deficiente para conseguir acceder a nuestra base de datos, obtener datos de ella, modificarlos o incluso eliminarlos. Mediante estos ataques también se pueden obtener accesos a partes de la aplicación web que deberían estar controlados a través de login.

Por los motivos antes descritos es de vital importancia que los desarrolladores escriban un código que impida la posibilidad de estos ataques y que garantice la integridad de los datos que hay en la BD.

La primera prueba que se va a realizar, es un simple intento de login sin poseer usuario en el sistema. Para esto vamos a intentar introducir código SQL en alguno de los campos con la intención de obtener privilegios y así poder acceder.

Ponemos en el campo de Username la siguiente cadena: *‘ or 1=1* – y intentamos hacer login.



En nuestro caso particular no ha funcionado. No hemos podido obtener acceso al sistema intentado modificar la sentencia de SQL que se envía a la base de datos cuando se hace un login.
Como se esperaba seguramente el programador de la web haya tenido en cuenta esta vulnerabilidad y haya programado la aplicación de forma que no se puedan conseguir accesos indeseados con facilidad.

Para la próxima prueba utilizaremos la herramienta sqlmap. Es una herramienta desarrollada en phyton y que sirve para realizar inyección de código sql. Tiene por objetivo detectar y aprovechar las vulnerabilidades de las aplicaciones web a la inyección de SQL.

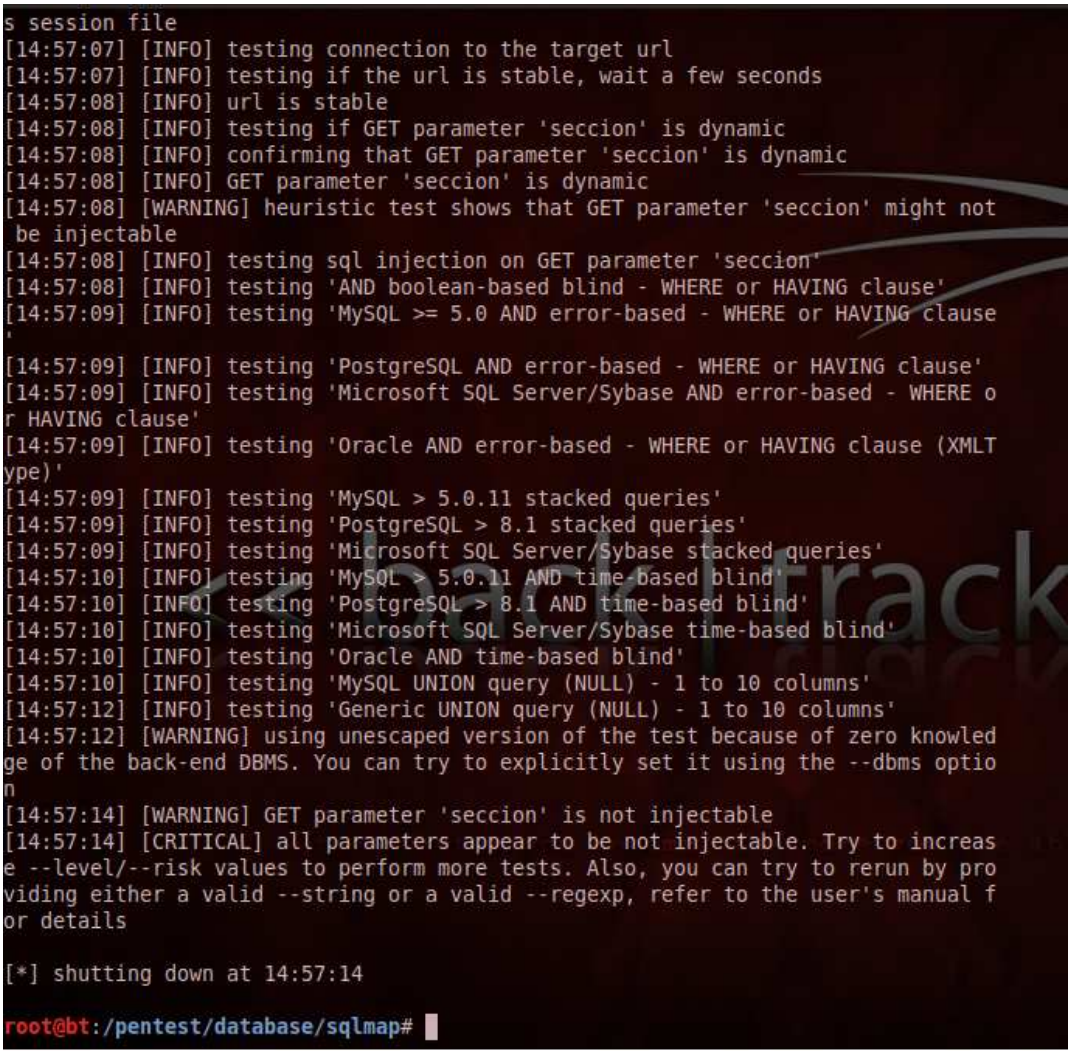
Esta prueba se ha realizado bajo un entorno de test. Se ha montado una máquina virtual con Backtrack 5 que es una distribución en formato live/cd diseñada para la auditoria y seguridad. Se ha montado ya que incluye la herramienta que vamos a utilizar.

Si no quisiésemos montar un entorno de prueba podríamos descargar la aplicación desde la página http://sqlmap.sourceforge.net/ e instalarla en nuestra máquina.

Para usar la herramienta sqlmap debemos escribir el siguiente comando:

$ ./sqlmap.py –u “dirección de la web que queremos atacar”

La web la tenemos alojada en la dirección del firewall que se encuentra en 192.168.1.77 provisionalmente.



Como vemos la aplicación ha intentando encontrar alguna vulnerabilidad mediante el testeo de las diferentes posibilidades de inyección SQL.

El no haber conseguido encontrar la forma de acceder mediante estás dos pruebas es un buen síntoma que nos indica que el desarrollado ha tenido en cuenta posibles ataques mediante la inyección de SQL.
Aunque no hayamos obtenido acceso no se puede garantizar que la página no sea vulnerable a otro tipo de ataques.

Es imprescindible que junto a una buena programación se realicen actualizaciones de seguridad necesarias para garantizar que el sistema siga siendo lo más seguro posible.

6.4 Pruebas vulnerabilidad Firewall y Apache

En este apartado se analiza el estado de nuestro sistema visto desde el exterior. Vamos a ver qué puertos son accesibles, que es lo que puede ver alguien que tiene intención de atacar alguno de los servicios o incluso que es lo que sucede cuando se realiza un ataque DDOS sobre el servidor apache.

Nmap es una herramienta muy potente que sirve para efectuar rastreo de puertos. Indicándole la máquina que se desea analizar nos da un pequeño informe sobre qué servicios son accesibles desde la red.

Vamos a ver cómo funciona con un ejemplo escaneando nuestro firewall.

nmap 192.168.1.77

```
Starting Nmap 5.21 ( http://nmap.org ) at 2012-06-18 00:48 CEST
Nmap scan report for 192.168.1.77
Host is up (0.0024s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP_address (1 host up) scanned in 5.54 seconds
```

El resultado nos indica que ha encontrado un puerto abierto, el 80. Si no indicamos ningún parámetro nmap no escanea todos los puertos posibles.

En la siguiente prueba vamos a indicarle que escanee los puertos asignados entre dos rangos.

nmap -p 1-30000 192.168.1.77

```
Starting Nmap 5.21 ( http://nmap.org ) at 2012-06-18 01:30 CEST
Nmap scan report for 192.168.1.77
Host is up (0.0019s latency).
Not shown: 29996 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
13000/tcp open  unknown
13001/tcp open  unknown
13002/tcp open  unknown
MAC Address: 00:04:75:D0:86:F4 (3 Com)
```

Ahora a parte de ver el puerto 80 que se había detectado en la fase anterior nos indica que hay 3 puertos más abiertos pero al no ser los puertos por defecto no sabe a que

servicio corresponde. Una de las ventajas de asignar al ssh puertos que no son el puerto por defecto es que se camufla un poco ante aplicaciones como nmap.

El servicio ssh nos da acceso a las máquinas de nuestra red. Es importante que las medidas de seguridad sean consideradas como prioritarias. Dado que es muy complejo proteger el sistema ante todos los posibles ataques se recomienda restringir el acceso de usuarios a este tipo de servicio. Si un atacante consiguiese saltarse la seguridad y acceder a una máquina mediante este servicio podría llegar a conseguir el control total sobre la máquina o incluso sobre otras máquinas conectadas a la misma red.
Es recomendable establecer passwords fuertes y cambiarlos con regularidad. También es muy aconsejable restringir el acceso al usuario root a través de ssh y vigilar los logs de las conexiones ssh.

Vamos a ver un ejemplo de un escaneo de puertos de un servidor donde el nivel de seguridad es bajo y del cual podemos obtener mucha información sobre que servicios tiene disponibles y a cuales se podría llegar a atacar.

nmap XXX.XXX.XXX.XXX

```
Not shown: 985 closed ports
PORT     STATE    SERVICE
21/tcp   open     ftp
22/tcp   open     ssh
25/tcp   open     smtp
53/tcp   open     domain
80/tcp   open     http
106/tcp  open     pop3pw
110/tcp  open     pop3
143/tcp  open     imap
443/tcp  open     https
445/tcp  filtered microsoft-ds
465/tcp  open     smtps
587/tcp  open     submission
993/tcp  open     imaps
995/tcp  open     pop3s
8443/tcp open     https-alt

Nmap done: 1 IP address (1 host up) scanned in 24.80 seconds
```

Nmap nos devuelve un listado de puertos abiertos y sus respectivos servicios. Esto podría utilizarse para ciertas vulnerabilidades si no se tiene en cuenta a la hora de configurar los servicios. Por ejemplo el servicio FTP que usa el puerto 21 es un tipo de servicio muy explotado para conseguir accesos a los sistemas de usuarios no autorizados.

Cada puerto abierto es una posibilidad de ataque. En el caso de que todos los servicios estuviesen centralizados en una sola máquina la posibilidad de recibir un ataque y que éste tenga éxito es bastante elevada. También hay que pensar que si consiguen el control de la máquina o tirar abajo un servicio como por ejemplo el servidor apache estaría afectando al resto de servicios y máquinas de la red.

Como ya se ha comentado al principio nmap es una aplicación muy potente con la que podemos obtener mucha información sobre cualquier máquina conectada a internet. Tiene muchas opciones y es recomendable echarle un vistazo para poder analizar a fondo el sistema.

Es importante realizar pruebas y asegurarnos de que la configuración que hemos establecido es la correcta para estar protegidos al mayor número de peligros posible. También debemos mantener nuestro software actualizado ya que muchas veces los bugs y vulnerabilidades conocidos se arreglan en las actualizaciones.

Una buena configuración de nuestro firewall nos puede proteger de ataques en tiempo real como se indica en el manual de la página: http://www.cyberciti.biz/faq/linux-detect-port-scan-attacks/
Básicamente nos explica que a través de la política drop se puede realizar un seguimiento mediante el envio de emails donde indicar posibles atacantes y sus ips, las cuales se pueden añadir a una lista de bloqueo para que no puedan acceder a nuestro servicio.

Para concluir este capitulo vamos a realizar una prueba sobre apache y su capacidad para aguantar posibles ataques.

Uno de los ataques más conocidos realizados sobre servidores web es el ataque DDOS o deny of services que consiste en hacer que un servicio o recurso sea inaccesible para los usuarios de éste.

Para realizar esta simulación vamos a utilizar el programa LOIC (Low Orbit Ion Cannon).

Es muy fácil de instalar y de utilizar. Tan solo debemos fijar la ip a la que queremos lanzar el ataque y apretar el botón de inicio para que se empiecen a enviar paquetes a la ip marcada como objetivo.

Vamos a ver una imagen del top del webserver para comprobar que realmente llegan los paquetes que se envían desde la aplicación LOIC.

```
top - 03:08:52 up 15:43,  3 users,  load average: 0.40, 0.10, 0.07
Tasks: 257 total,   1 running, 256 sleeping,   0 stopped,   0 zombie
Cpu(s):  5.6%us, 18.5%sy,  0.0%ni, 72.9%id,  0.0%wa,  0.0%hi,  3.0%si,  0.0%s
Mem:    247404k total,   243384k used,     4020k free,     1912k buffers
Swap:   518140k total,     4336k used,   513804k free,    39656k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
  931 root      20   0 33412 5576 3568 S  3.6  2.3   0:01.70 apache2
   23 root      20   0     0    0    0 S  3.3  0.0   0:00.39 kswapd0
 4728 webserve  20   0  2632 1112  840 R  0.7  0.4   0:00.78 top
 4729 www-data  20   0 33820 4260 2000 S  0.7  1.7   0:00.40 apache2
 4730 www-data  20   0 33820 4260 2000 S  0.7  1.7   0:00.40 apache2
    3 root      20   0     0    0    0 S  0.3  0.0   0:00.36 ksoftirqd/0
 1293 www-data  20   0 33908 4756 2164 S  0.3  1.9   0:00.17 apache2
 4294 root      20   0     0    0    0 S  0.3  0.0   0:00.91 kworker/0:1
 4611 root      20   0     0    0    0 S  0.3  0.0   0:00.02 kworker/0:2
 4630 webserve  20   0  8956 1576  928 S  0.3  0.6   0:00.03 sshd
 4731 www-data  20   0 33820 4380 2076 S  0.3  1.8   0:00.19 apache2
 4741 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.05 apache2
 4751 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4756 www-data  20   0 33748 4356 2076 S  0.3  1.8   0:00.02 apache2
 4759 www-data  20   0 34036 4376 2076 S  0.3  1.8   0:00.23 apache2
 4762 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4765 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4769 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4771 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.05 apache2
 4773 www-data  20   0 34036 4376 2076 S  0.3  1.8   0:00.24 apache2
 4800 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4808 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4814 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4817 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4819 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4821 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4822 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.02 apache2
 4823 www-data  20   0 33748 4344 2072 S  0.3  1.8   0:00.09 apache2
 4827 www-data  20   0 34036 4376 2076 S  0.3  1.8   0:00.23 apache2
 4833 www-data  20   0 33748 4352 2072 S  0.3  1.8   0:00.05 apache2
 4836 www-data  20   0 33412 2728  668 S  0.3  1.1   0:00.01 apache2
```

Se puede observar que realmente el servidor apache está empezando a sufrir con tanto proceso. Aunque esto pueda parecer contrario apache es uno de los servidores web más robustos en la actualidad. Es bastante complejo tirar un servidor con una sola máquina realizando este tipo de ataques pero eso no significa que aunque no pueda denegarnos el servicio no provoque que los tiempos de respuesta aumenten considerablemente.

En todas la pruebas que se han realizado con la aplicación LOIC no se ha conseguido tirar abajo el servidor apache. Se ha probado a realizar múltiples ataques desde 2 máquinas y el servidor ha conseguido aguantar. Es cierto que si varias máquinas realizasen este tipo de ataque acabarían por saturar nuestro servidor ya que solo dispone de 256MB de RAM y al tener que servir tantas peticiones seguramente se quedaría sin memoria libre.

En la actualidad se pueden evitar este tipo de ataques desde la configuración de iptables.

Bastaría con añadir a nuestro script una línea que limitase el número de paquetes que pueden llegar al servicio de la web y de esta forma estaríamos consiguiendo minimizar el número de peticiones que este tipo de aplicaciones podrían provocar.

La siguiente línea

```
iptables -A INPUT -p tcp --dport 80 -m limit --limit 50/minute --
limit-burst 100 -j ACCEPT
```

Nos permite filtrar el número de paquetes por minuto para evitar posibles ataques DDOS.

```
top - 03:18:19 up 15:52,  4 users,  load average: 0.01, 0.03, 0.05
Tasks: 189 total,   1 running, 188 sleeping,   0 stopped,   0 zombie
Cpu(s):  3.3%us,  1.0%sy,  0.0%ni, 95.3%id,  0.3%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:    247404k total,   236404k used,    11000k free,     6208k buffers
Swap:   518140k total,     7964k used,   510176k free,    55344k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2302	root	20	0	58028	23m	4380	S	2.0	9.7	0:24.70	Xorg
5407	webserve	20	0	2632	1168	852	R	0.7	0.5	0:00.07	top
2560	webserve	20	0	86828	12m	9152	S	0.3	5.0	0:03.47	gnome-terminal
4973	webserve	20	0	2632	1288	924	S	0.3	0.5	0:02.27	top
5135	www-data	20	0	33748	3776	2072	S	0.3	1.5	0:00.04	apache2
5140	www-data	20	0	34036	3792	2076	S	0.3	1.5	0:00.06	apache2
5207	www-data	20	0	34036	3792	2076	S	0.3	1.5	0:00.03	apache2
5208	www-data	20	0	33792	3788	2076	S	0.3	1.5	0:00.04	apache2
5249	www-data	20	0	33748	3776	2072	S	0.3	1.5	0:00.03	apache2
5260	www-data	20	0	33748	3776	2072	S	0.3	1.5	0:00.02	apache2
1	root	20	0	3044	1556	1040	S	0.0	0.6	0:01.09	init
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.44	ksoftirqd/0
5	root	20	0	0	0	0	S	0.0	0.0	0:00.55	kworker/u:0
6	root	RT	0	0	0	0	S	0.0	0.0	0:00.00	migration/0
7	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	cpuset
8	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	khelper
9	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	netns
10	root	20	0	0	0	0	S	0.0	0.0	0:00.10	sync_supers
11	root	20	0	0	0	0	S	0.0	0.0	0:00.00	bdi-default
12	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kintegrityd
13	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kblockd
14	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kacpid
15	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kacpi_notify
16	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kacpi_hotplug
17	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	ata_sff
18	root	20	0	0	0	0	S	0.0	0.0	0:00.00	khubd
19	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	md
22	root	20	0	0	0	0	S	0.0	0.0	0:00.02	khungtaskd
23	root	20	0	0	0	0	S	0.0	0.0	0:00.62	kswapd0
24	root	25	5	0	0	0	S	0.0	0.0	0:00.00	ksmd

Realizando los mismos ataques hemos conseguido reducir el número de paquetes que llegan a nuestro servidor. El único problema que podría ocasionarnos ahora un ataque de este tipo seria la posibilidad de saturar el firewall y producir un cuello de botella. Mientras que el servidor apache aguantaría y seguiría dando servicio nuestro firewall quedaría saturado.

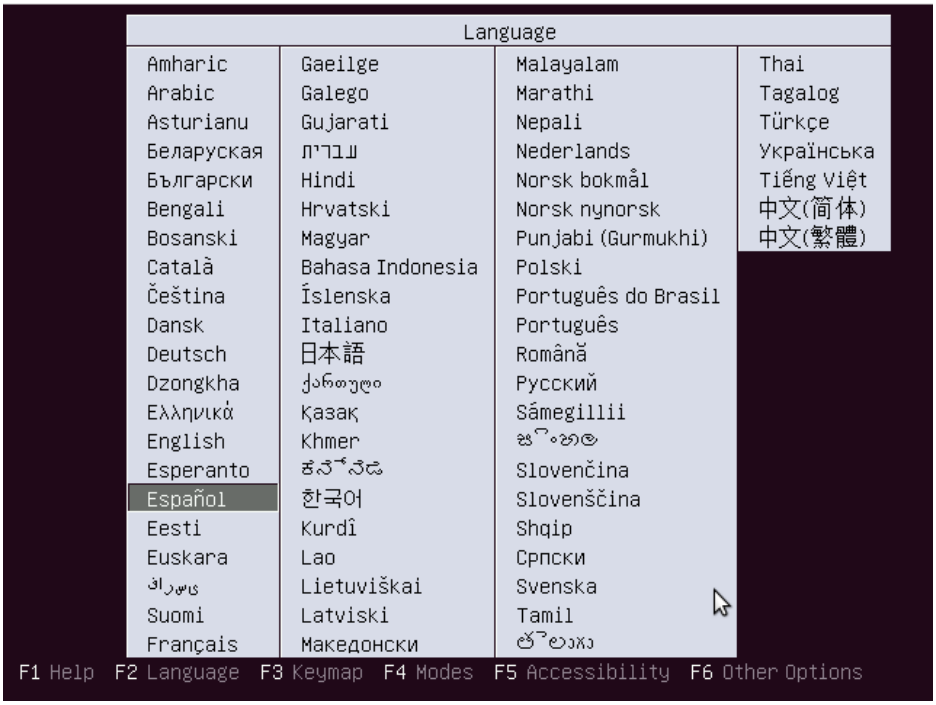
7. Manual de Instalación y configuración

7.1 Instalación Ubuntu Server 11.04

La instalación de Ubuntu Server se ha hecho a partir de un cd. Se ha descargado una iso de la página oficial de Ubuntu y se ha grabado en un cd. Se podría realizar la instalación usando un usb pero las máquinas de las que se dispone son muy antiguas y no se puede realizar la instalación a partir de ese dispositivo ya que el hardware no permite arrancar desde usb al no tener los drivers instalados.

La versión descargada de Ubuntu es la de 32bits.

Una vez insertamos el cd y configuramos la máquina para que arranque desde la unidad de cd aparecerá la siguiente pantalla



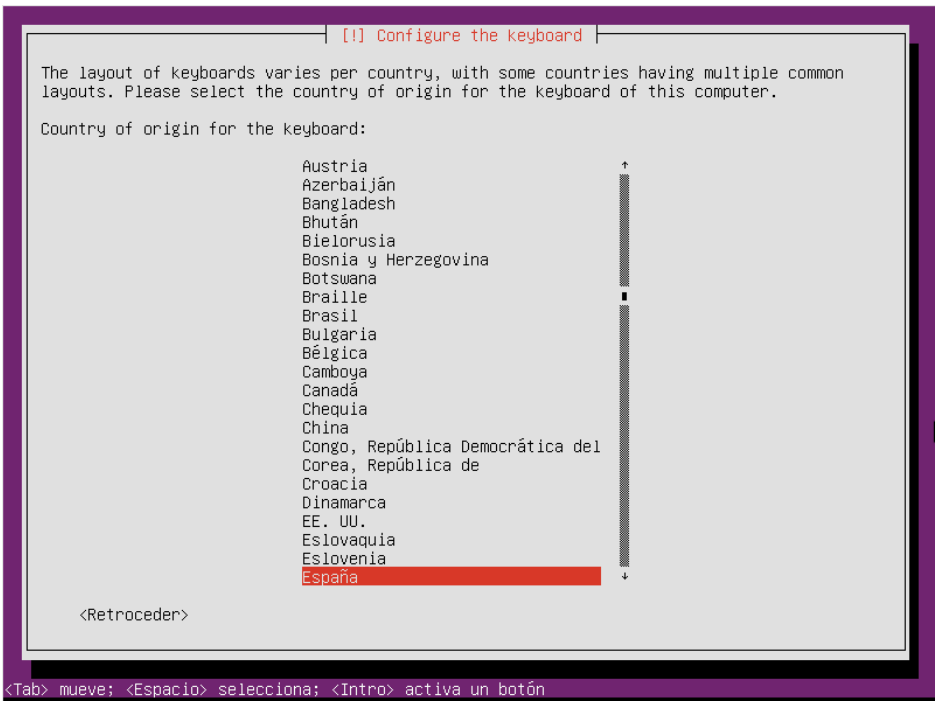
Aquí seleccionamos el idioma en el que queremos instalar el SO y apretamos enter.

Se pueden mirar otras configuraciones antes de empezar la instalación en los submenús que aparecen en el inferior.

El siguiente paso es seleccionar que acción queremos realizar, en nuestro caso seleccionamos la opción de Instalar Ubuntu Server en modo normal.

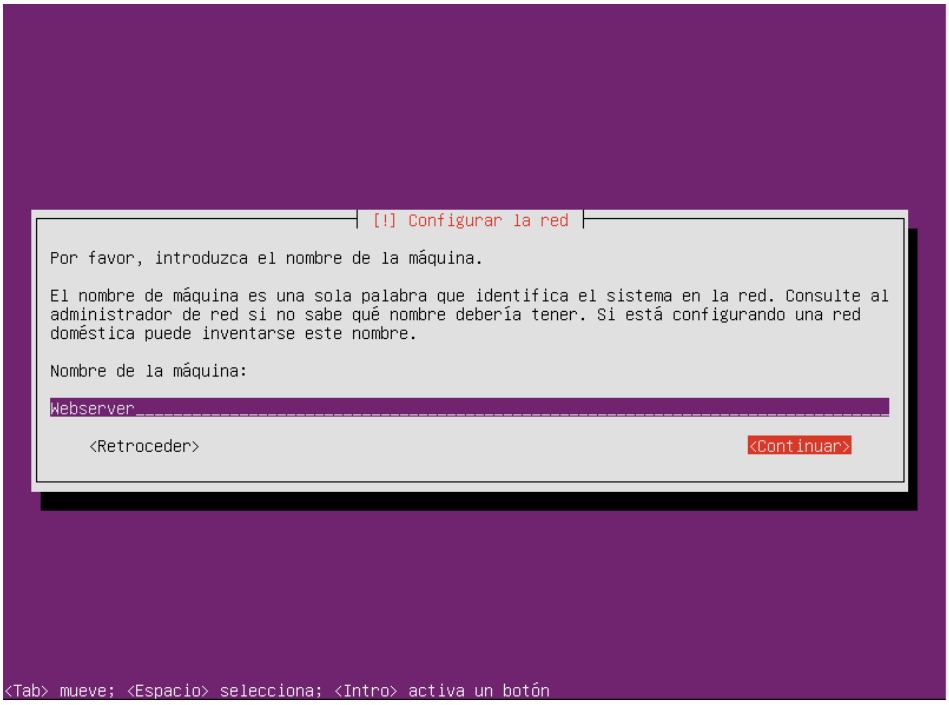
Una vez iniciada la instalación nos aparece una pantalla donde seleccionamos nuestra ubicación y continuamos.

En el siguiente paso nos preguntan sobre la configuración del teclado. Nos consulta si queremos detectar la disposición del teclado y nosotros seleccionamos la opción de ‘no’ ya que en la siguiente pantalla podemos seleccionar la configuración del teclado por país. Así que en la siguiente pantalla seleccionamos que la configuración del teclado por defecto debe ser la de España. Esta configuración del teclado se puede cambiar una vez acabada la instalación por lo que tampoco es demasiado importante si nos equivocamos o elegimos una configuración diferente a la que realmente tenemos.

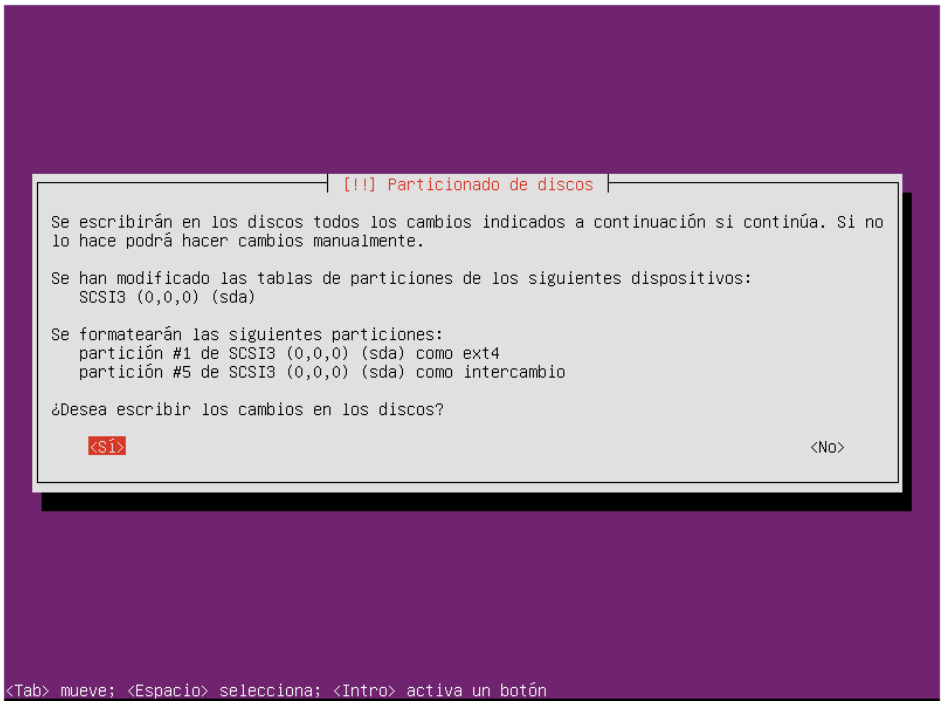


Nos pide que variante queremos del teclado y seleccionamos la primera, la de España sin variantes y con todas las teclas disponibles.

Lo siguiente será poner nombre a nuestra máquina y darle a continuar.



En la siguiente pantalla nos pide si queremos configurar el reloj y nos pregunta si la zona horaria corresponde al país elegido, seleccionamos que si y continuamos.
A continuación viene el paso más importante de la instalación donde indicaremos que proporción de disco duro queremos utilizar y que particiones se debe realizar.
Indicamos que queremos un método de partición guiado y que queremos utilizar todo el disco.
Nos aparece la siguiente pantalla



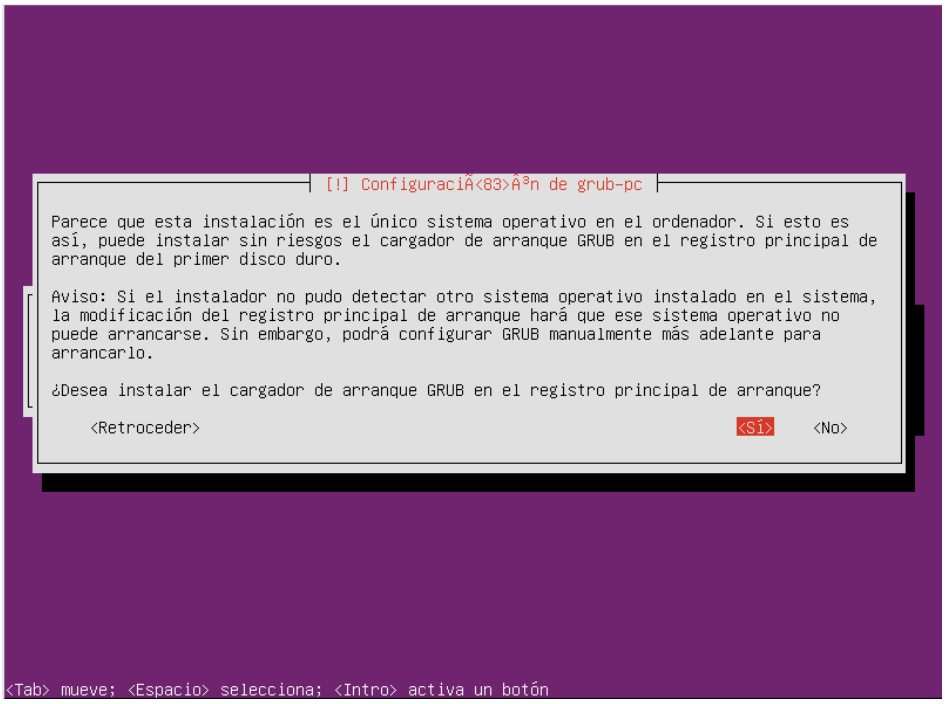
Deben aparecer dos particiones una para swap o intercambio y otra donde irá la raíz /.
Al aceptar empezará a formatearse el disco duro y se crearán las particiones que se nos ha indicado en la pantalla de confirmación.
Acabado el formateo y partición del disco se inicia la instalación del sistema base. Este proceso puede durar algunos minutos.
El siguiente paso es la configuración de un usuario para no usar la cuenta de superusario.

En las siguientes pantallas se nos pide que elijamos una contraseña para el nuevo usuario. Introducimos el password y confirmamos.
Seguidamente se nos pregunta si queremos cifrar la carpeta personal del usuario y le indicamos que no queremos cifrar la carpeta personal del nuevo usuario.

La instalación nos pregunta si tenemos que utilizar un proxy HTTP para acceder a la red. Dejamos el campo en blanco y presionamos el botón de continuar y nos lleva a la pantalla de configuración de actualizaciones. Indicamos que no deseamos actualizaciones automáticas y continuamos.

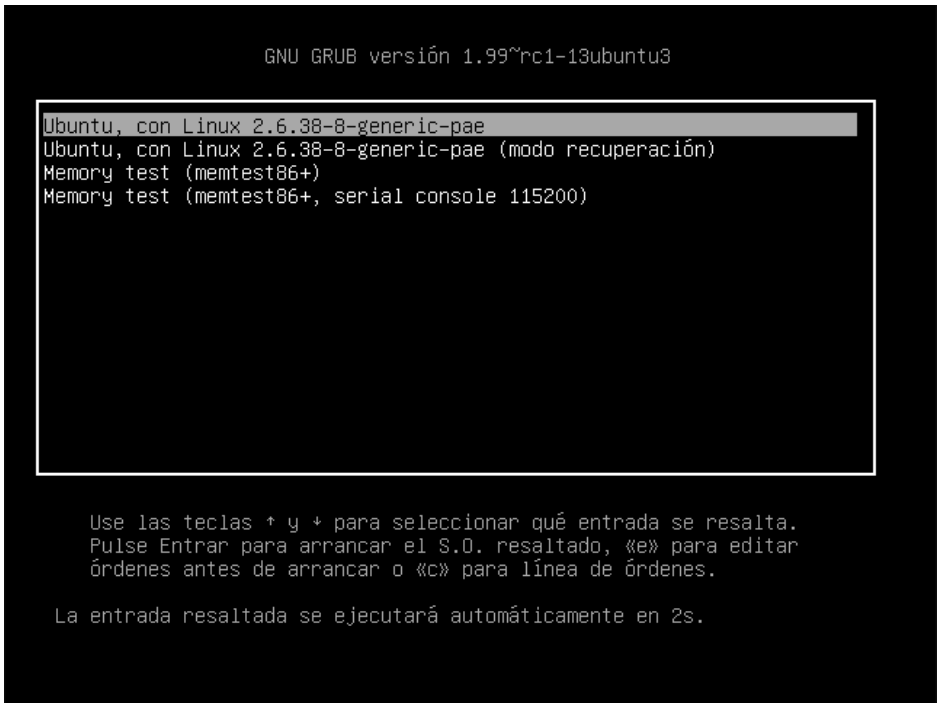
A continuación aparece una pantalla donde se nos pregunta si deseamos instalar alguna aplicación. Se podría seleccionar el paquete de openssh pero lo instalaremos posteriormente de forma manual. Pasamos a la siguiente pantalla sin seleccionar ningún paquete.

El último paso que queda por realizar es la configuración del gestor de arranque. Debemos instalarlo para que podamos iniciar la nueva instalación con normalidad.



Indicamos que si queremos instalarlo y aceptamos. Con este último paso hemos completado la instalación, nos aparecerá una pantalla que así nos lo indica. Debemos retirar el cd de instalación y a continuación se reiniciará el ordenador.

Al arrancar aparece el gestor de arranque grub donde podremos seleccionar que tipo de arranque deseamos lanzar.



7.2 Configuración de la red

Para poder configurar la red primero debemos saber de que dispositivos de red dispone la máquina. Utilizaremos el comando 'lspci' y filtraremos todos los mensajes referentes a tarjetas de red de la siguiente forma:

```
$ lspci | grep Ethernet
```

Obtendremos algo parecido a esto:

```
firewall@firewall:~$ lspci | grep Ethernet
01:00.0 Ethernet controller: D-Link System Inc RTL8139 Ethernet (rev 10)
01:01.0 Ethernet controller: 3Com Corporation 3c905C-TX/TX-M [Tornado] (rev 78)
01:03.0 Ethernet controller: 3Com Corporation 3c905C-TX/TX-M [Tornado] (rev 78)
01:08.0 Ethernet controller: Intel Corporation 82801DB PRO/100 VE (LOM) Etherne
 Controller (rev 81)
firewall@firewall:~$
```

Una vez sepamos que tarjetas de red vamos a configurar debemos editar el archivo *'interfaces'* que se encuentra en *'/etc/network/'*

```
$ sudo vi /etc/network/interfaces
```

Cada servidor tiene su archivo de configuración de interfaces que debe quedar como se muestra a continuación:

- Webserver:

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.30.5
netmask 255.255.255.0
gateway 192.168.30.7
```

- Firewall:

```
auto lo
iface lo inet loopback

# Tarjeta exterior
auto eth0
iface eth0 inet static
address 84.88.32.226
netmask 255.255.255.0
gateway 84.88.32.225
#Tarjeta BD
```

```
auto eth1
iface eth1 inet static
address 10.10.1.5
netmask 255.0.0.0

#Tarjeta Webserver
auto eth2
iface eth2 inet static
address 192.168.30.7
netmask 255.255.255.0
```

- Servidor BD:

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 10.10.1.2
netmask 255.0.0.0
gateway 10.10.1.5
```

En nuestro caso en particular en los servidores de base de datos y webserver configuramos una tarjeta de red. En el firewall configuramos tres tarjetas que serán conectadas a las diferentes zonas de nuestra red (DMZ, lan y exterior).

Para actualizar los cambios debemos reiniciar el servicio de red mediante el siguiente comando:

```
$ sudo /etc/init.d/networking restart
```

```
webserver@Webserver:~$ sudo /etc/init.d/networking restart
 * Running /etc/init.d/networking restart is deprecated because it may not enable again some interfaces
 * Reconfiguring network interfaces...
ssh stop/waiting
ssh start/running, process 3555
```

El siguiente paso es configurar el servicio de DNS en el firewall ya que será el único servidor con acceso a internet. Editaremos el archivo '*resolv.conf*', si no existe lo crearemos.

```
$ sudo vi /etc/resolv.conf
```

Añadimos los servidores de DNS que vamos a utilizar:

```
nameserver 8.8.8.8
nameserver 147.83.38.14
nameserver 147.83.2.3
```

7.3 Actualización repositorios

Para poder instalar las últimas versiones de los paquetes desde los diferentes repositorios nos tenemos que asegurar de que estos estén actualizados. Para ello utilizaremos el siguiente comando:

```
$ sudo apt-get update
```

A veces este proceso de actualización se puede quedar bloqueado dando paso al error "E:No se pudo bloquear /var/lib/apt/list/lock…"
Para solventar este problema basta con borrar el archivo 'lock' que es el encargado de bloquear los repositorios cuando se están actualizando.

```
$ sudo rm /var/lib/apt/lists/lock
```

Si necesitásemos agregar algún repositorio a nuestra lista bastaría con hacerlo de la siguiente manera:

```
$ sudo add-apt-repository repositorios
```

7.4 Instalación entorno gráfico

Se ha instalado un entorno gráfico ligero y con los componentes básicos para que haya menos aplicaciones que puedan producir vulnerabilidades.

Para instalar las x primero debemos instalar el paquete xserver-xorg.

```
$ sudo apt-get install xserver-xorg
```

Una vez instalado el servidor del entorno grafico instalamos la versión ligera de gnome

```
$ sudo apt-get install xorg gnome-core
```

Para poder arrancar el entorno gráfico escribimos la siguiente instrucción

```
$ startx
```

7.5 Instalación Apache, Php y aplicación web

Para poder montar y configurar el webserver tendremos que montar un servidor web donde alojar la página web. Para ello instalaremos el servidor HTTP Apache y el modulo de php para que las páginas programadas en este lenguaje funcionen en nuestro servidor.

Instalación servidor Apache

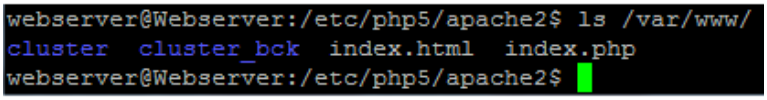
```
$ sudo apt-get install apache2
```

Instalación php5

```
$ sudo apt-get install php5
$ sudo apt-get install php5-mysql
```

Las páginas las alojaremos en el sitio por defecto de apache que se encuentra en *'/var/www/'*, allí copiaremos todos los archivos de nuestra página dentro de una carpeta llamada, por ejemplo, *'cluster'*.

```
$ ls /var/www
```



El siguiente paso que vamos a realizar es crear un virtualhost. Con los virtualhost podemos tener varios dominios apuntando a la misma máquina, es decir que podríamos tener varias páginas alojadas en el mismo servidor.
Para crearlo debemos crear un archivo en la carpeta *'/etc/apache2/sites-available'* que es la carpeta donde se encuentran todas las webs disponibles.

```
$ cd /etc/apache2/sites-available
$ sudo cat default > cluster
$ sudo vi cluster
```

Una vez hayamos copiado el archivo entramos para modificar el contenido de éste y lo dejamos como se detalla a continuación:

```
<VirtualHost *:80>
    ServerAdmin webmaster@cluster.com
    ServerName cluster.com
    ServerAlias www.cluster.com
    DocumentRoot /var/www/cluster
    <Directory /var/www/cluster>
        AllowOverride All
        Allow from All
```

```
        </Directory>
        CustomLog ${APACHE_LOG_DIR}/cluster.com-access.log combined
    </VirtualHost>
```

```
 GNU nano 2.2.6   Archivo: /etc/apache2/sites-available/cluster

<VirtualHost *:80>
        ServerAdmin webmaster@localhost
        ServerName cluster.com
        ServerAlias www.cluster.com

        DocumentRoot /var/www/cluster

        <Directory /var/www/cluster>
                AllowOverride All
                Allow from All
        </Directory>

        CustomLog ${APACHE_LOG_DIR}/cluster.com-access.log combined
</VirtualHost>
```

Toda la configuración de carpetas, nombres y puertos que puede mostrar el servidor se establecen en los archivos que hay en la carpeta de *'/etc/apache2/sites-available'*.

El DocumentRoot indica cual será la carpeta del sitio que queremos virtualizar.

En el directorio *'etc/apache2/sites-enabled'* se encuentran los softlinks a los sitios que están habilitados, son enlaces simbólicos a los archivos que se encuentran en la carpeta *'etc/apache2/sites-available'*.

Para crear el softlink utilizaremos el siguiente comando

```
$ sudo a2ensite cluster
```

Ahora ya tenemos el softlink al archivo existente en *sites-enabled*

```
$ ls -la /etc/apache2/sites-enabled
```

```
webserver@Webserver:/etc/php5/apache2$ ls -la /etc/apache2/sites-enabled
total 8
drwxr-xr-x 2 root root 4096 2012-06-05 22:12 .
drwxr-xr-x 7 root root 4096 2011-10-25 22:05 ..
lrwxrwxrwx 1 root root   26 2011-12-11 19:46 cluster -> ../sites-available/clust
er
webserver@Webserver:/etc/php5/apache2$
```

En resumen en la carpeta *'/etc/apache2/sites-enabled'* tenemos los sitios que apache puede utilizar y en *'/etc/apache2/sites-availabe'* los sitios disponibles.

Si quisiéramos deshabilitar un sitio utilizaríamos el siguiente comando

```
$ sudo a2disssite cluster
```

Este comando borra el softlink creado anteriormente y la página web no estaría disponible en el servidor apache.

Vamos a desactivar el sitio por defecto de apache

```
$ sudo a2disssite 000-default
```

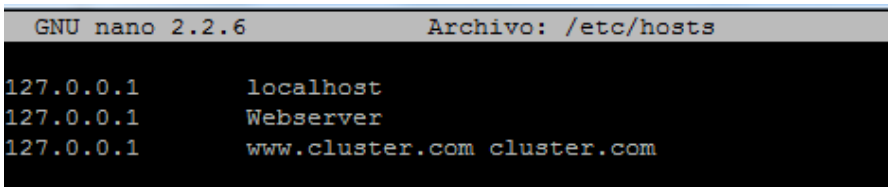
Ahora cada petición que le llegue a apache la redirigirá a la página del clúster.

A continuación hay que modificar el archivo '/etc/hosts'

```
$ sudo nano /etc/hosts
```

Y agregamos la siguiente línea '127.0.0.1 www.cluster.com cluster.com'

```
127.0.0.1       localhost
127.0.0.1       www.cluster.com cluster.com
```



Debemos activar en nuestro servidor el modulo '*Rewrite*' ya que la página lo necesita.

Este modulo se encarga de traducir url amigables (digamos que más sencillas) en urls más complejas que pueden incluir el paso de parámetros pero que a la vista del internauta son invisibles.

Para activar módulos en apache se utiliza el comando '*a2enmod*'.

```
$ sudo a2enmod rewrite
```

Si quisiéramos desactivarlo haríamos lo mismo pero utilizando el comando '*a2dismod*'.

También debemos activar el uso de acentos y caracteres especiales para que estos se vean de forma correcta en la página. Debemos modificar el archivo 'charset' que se encuentra en *'/etc/apache2/conf.d/'*.

```
$ sudo nano /etc/apache2/conf.d/charset
```

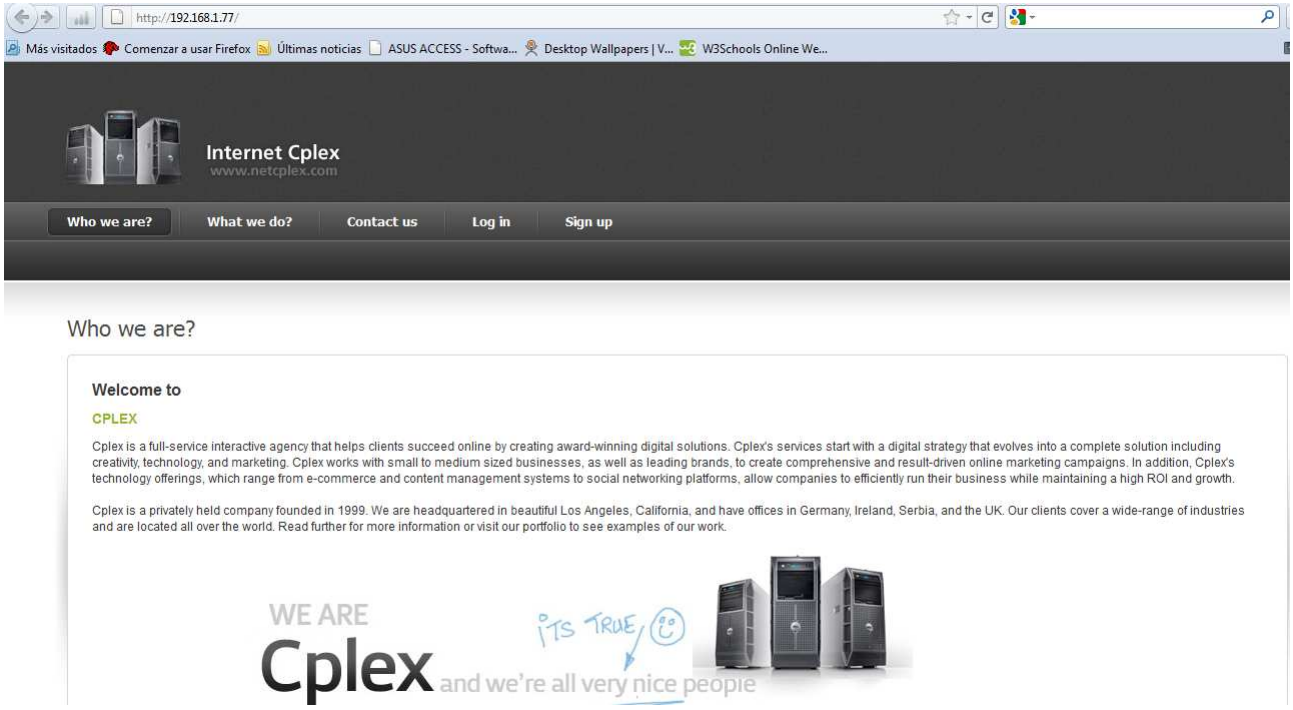
Descomentar la línea:

```
AddDefaultCharset   UTF8
```

Por último reiniciamos el servidor apache con el siguiente comando

```
$ sudo /etc/init.d/apache2 restart
```

Para comprobar que tenemos acceso a la página podemos realizar una prueba desde nuestro navegador.



Para terminar debemos configurar los parámetros de conexión a la base de datos de la página. El archivo se encuentra en la carpeta '/var/www/cluster/Connections'.

```
$ sudo nano /var/www/cluster/Connections/cluster.php
```

```
<?php
# FileName="Connection_php_mysql.htm"
# Type="MYSQL"
# HTTP="true"
$hostname_cluster = "10.10.1.2";
$database_cluster = "ClusterMngr";
$username_cluster = "cluster";
$password_cluster = "cluster";
$cluster = mysql_pconnect($hostname_cluster, $username_cluster, $password_cluster)
or trigger_error(mysql_error(),E_USER_ERROR);
?>
```

```
 GNU nano 2.2.6                    Archivo: /var/www/cluster/Connections/cluster.php

<?php
# FileName="Connection_php_mysql.htm"
# Type="MYSQL"
# HTTP="true"
#$hostname_cluster = "192.168.30.5";
$hostname_cluster = "10.10.1.2";
$database_cluster = "ClusterMngr";
$username_cluster = "cluster";
$password_cluster = "cluster";
$cluster = mysql_pconnect($hostname_cluster, $username_cluster, $password_cluster) or trigger_error(mysql_error(),E_USER_ERROR);
?>
```

7.6 Instalación MySql

En el servidor de base de datos instalaremos principalmente mysql, un sistema de ficheros compartidos (nfs) y las aplicaciones java encargadas de controlar todos los jobs del cluster (manager, pendinginfo, agents…).

Instalación de mysql:

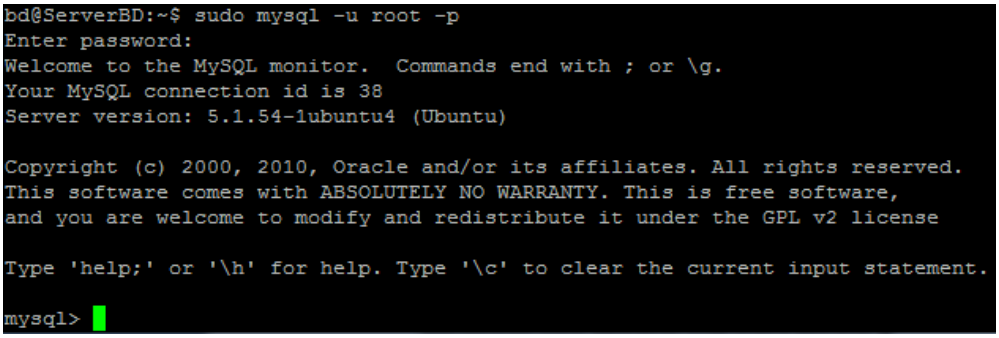
```
$ sudo apt-get install mysql-server
```

Durante la instalación se nos pedirá un password para el usuario root con el que posteriormente podremos acceder a mysql.

```
User: root
Password: root
```

Una vez instalado mysql podemos acceder a este mediante el comando:

```
$ sudo mysql -u root -p
```



Ahora ya estamos dentro de la consola de mysql y podemos ejecutar todas las acciones relacionadas con este gestor de base de datos.

Los scripts que contienen la base de datos y las tablas que necesitamos para el clúster necesitan una librería udf para mysql.
Hemos de copiar el archivo *'lib_mysqludf_sys.so'* que se encuentra en el cd del proyecto del clúster de computación en la carpeta *'/usr/lib/mysql/pluggin/'*.

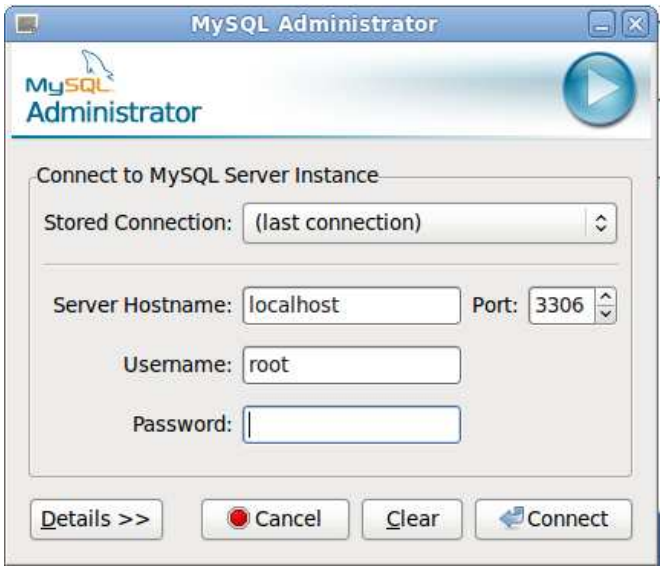
```
$ sudo cp lib_mysqludf_sys.so /usr/lib/mysql/pluggin
```

El siguiente paso será instalar un administrador de mysql que nos facilitará el trabajo para cuando queramos pasar los diferentes scripts.

```
$ sudo apt-get install mysql-admin
```

Una vez instalado el admin de mysql podemos acceder a éste mediante el comando

$ sudo mysql-admin



Creamos la base de datos pasando los scripts del clúster que encontramos en el cd de instalación del clúster. Los scripts que debemos ejecutar son:

ClusterMngr_skeleton_20110602.sql
ClusterMngr_initialData_20110602.sql
ClusterMngr_datosFacturacion.sql.sql

Podemos utilizar la herramienta MySql Query Browser o bien ejecutar los scripts entrando en la consola de MySql.

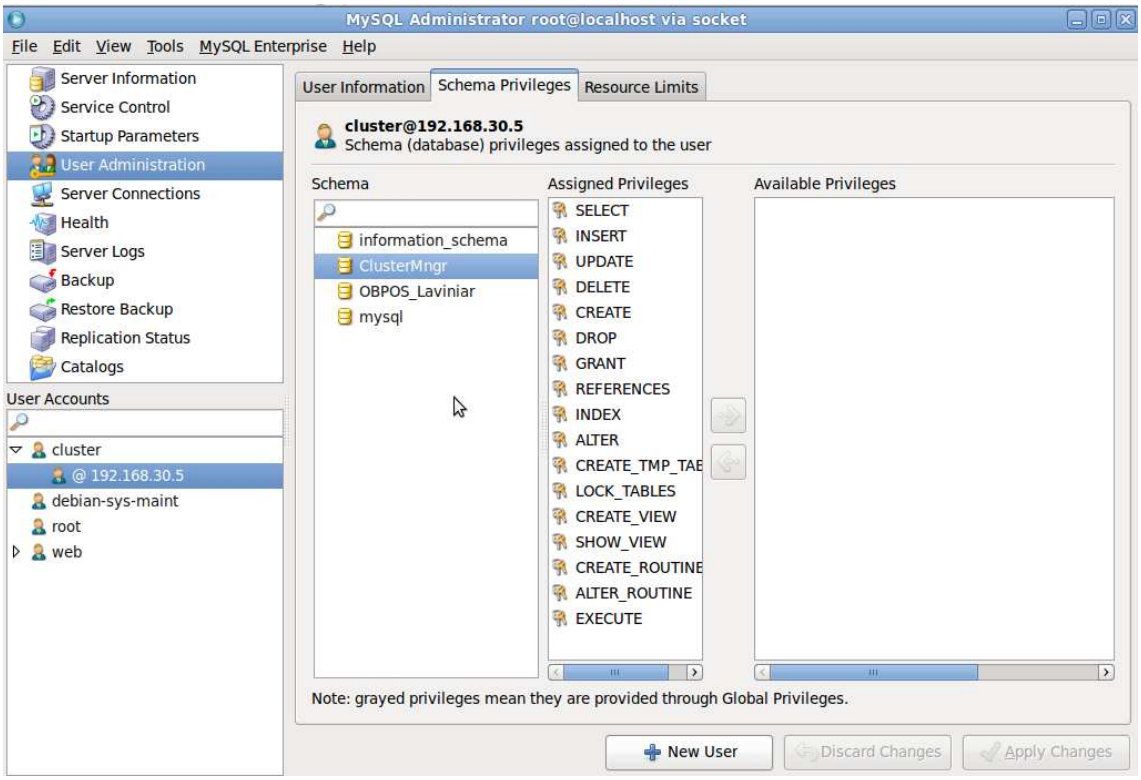
Una vez hayamos creado la estructura de la BD del clúster creamos un usuario para que se pueda conectar a la base de datos desde la web. Hay que crear un usuario para que se tenga acceso desde la DMZ. Entramos en la consola de Mysql.

$ sudo mysql -u root -p

Ejecutamos la siguiente instrucción.

> GRANT ALL PRIVILEGES ON cluster. TO cluster@192.168.30.5 IDENTIFIED BY 'cluster' WITH GRANT OPTION;*

También podemos hacerlo desde la herramienta *mysql-admin.*



Accedemos al menú User administrator y una vez allí creamos un usuario para nuestra bd.
En nuestro caso le hemos dado permiso solo a la base de datos del clúster para que sea mucho más seguro.

Para reiniciar el servicio de mysql utilizaremos la instrucción

$ sudo /etc/init.d/mysql restart

```
bd@ServerBD:~/scripts_BD$ sudo /etc/init.d/mysql restart
Rather than invoking init scripts through /etc/init.d, use the service(8)
utility, e.g. service mysql restart

Since the script you are attempting to invoke has been converted to an
Upstart job, you may also use the stop(8) and then start(8) utilities,
e.g. stop mysql ; start mysql. The restart(8) utility is also available.
mysql stop/waiting
mysql start/running, process 3806
bd@ServerBD:~/scripts_BD$
```

7.7 Instalación NFS

Para instalar el sistema de archivos NFS y poder compartir archivos entre diferentes máquinas primero tendremos elegir en que máquina irá instalado el servidor. En nuestro caso lo vamos a instalar en la máquina donde se encuentra la aplicación del clúster.

Primero vamos a configurar la parte del server.

```
$ sudo apt-get install nfs-kernel-server
```

NFS utiliza un servicio llamado portmap, lo instalamos con el siguiente comando

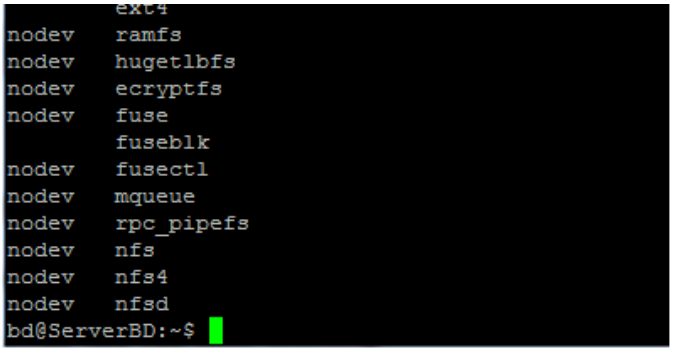
```
$ sudo apt-get install portmap
```

También instalamos el paquete para clientes de nfs

```
$ sudo apt-get install nfs-common
```

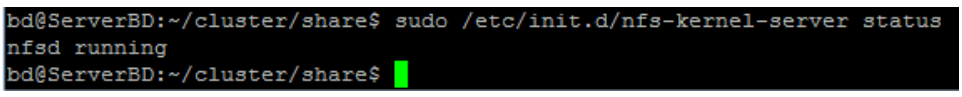
Si hacemos un cat del archivo */proc/filesystems* podemos observar que se ha instalado el server.

```
$ cat /proc/filesystem
```



Con los siguientes comandos podremos arrancar, parar, reiniciar y ver el status del servidor.

```
$ sudo /etc/init.d/nfs-kernel-server start
$ sudo /etc/init.d/nfs-kernel-server stop
$ sudo /etc/init.d/nfs-kernel-server restart
$ sudo /etc/init.d/nfs-kernel-server status
```

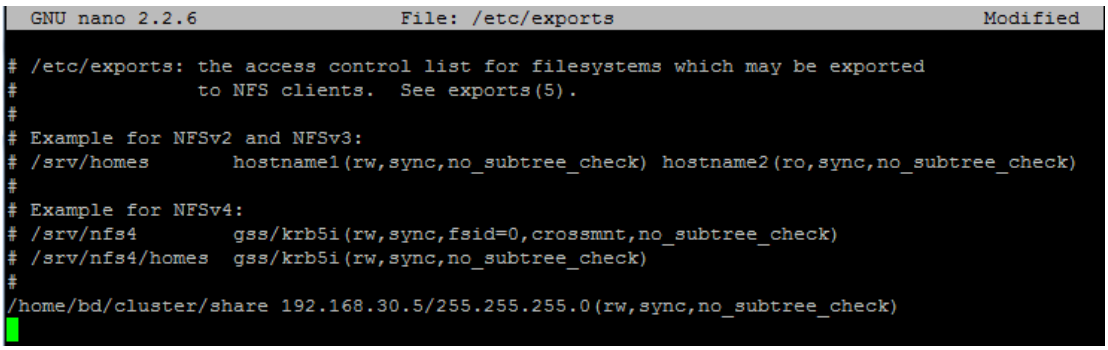


Los archivos compartidos se indican en la carpeta /etc/exports y es ahí donde tendremos que añadir una línea con el recurso que queramos compartir. Vamos a compartir la carpeta share que se encuentra en /home/bd/cluster/share.

$ sudo nano /etc/exports

Añadimos la siguiente línea:

/home/bd/cluster/share 192.168.30.5/255.255.255.0(rw,sync,no_subtree_check)



La ip que se indica corresponde a la del servidor web que es al que queremos darle permisos para poder acceder a la carpeta share. Le ponemos los flags correspondientes para que pueda a parte de leer, escribir. También indicamos que los recursos se deben sincronizar cada vez que se produzca algún cambio.

Los flags que podemos utilizar son:

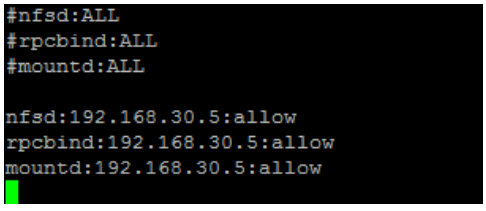
ro|rw: read only | read and write
async | sync: asícrono | sícrono
no_root_squash | root_squash | all_squash
secure | insecure

El siguiente paso es configurar las opciones de seguridad para asegurarnos de que solo se cargue el recurso en la máquina deseada.
Los archivos que debemos editar son /etc/hosts.allow y hosts.deny

En hosts.allow añadimos las siguientes líneas

$ sudo nano /etc/hosts.allow

nfsd:192.168.30.5:allow
rpcbind:192.168.30.5:allow
mountd:192.168.30.5:allow



Y en hosts.deny debemos indicar que máquinas no podrán acceder a los servicios.

$ sudo nano /etc/hosts.deny

nfsd:ALL
rpcbind:ALL
mountd:ALL

```
nfsd:ALL
rpcbind:ALL
mountd:ALL
```

Cada vez que se realice un cambio en los archivos de exports, hosts.deny o hosts.allow debemos reiniciar el servidor nfs.

$ sudo /etc/init.d/nfs-kernel-server restart

```
bd@ServerBD:~$ sudo /etc/init.d/nfs-kernel-server restart
 * Stopping NFS kernel daemon                                    [ OK ]
 * Unexporting directories for NFS kernel daemon...              [ OK ]
 * Exporting directories for NFS kernel daemon...                [ OK ]
 * Starting NFS kernel daemon                                    [ OK ]
bd@ServerBD:~$
```

Es importante asegurarnos que la carpeta que queremos compartir dispone de los permisos necesario y que el propietario de ésta no sea root ya que sino el servidor destino no podría escribir en ella.

```
bd@ServerBD:~/cluster$ ls -la
total 24
drwxr-xr-x  6 bd bd 4096 2012-06-16 17:13 .
drwxr-xr-x 25 bd bd 4096 2012-06-13 09:01 ..
drwxr-xr-x  2 bd bd 4096 2012-06-13 09:01 agents
drwxr-xr-x  2 bd bd 4096 2012-06-13 09:01 manager
drwxr-xr-x  2 bd bd 4096 2012-06-13 09:01 pendinginfo
drwxr-xr-x  3 bd bd 4096 2012-06-17 19:21 share
```

El último paso es obligar a que el servicio mountd utilice un puerto fijo. Por defecto el portmapper asigna un puerto libre aleatorio al servicio mountd. Nosotros deberemos fijarlo ya que si no lo hacemos al ser aleatorio no podríamos abrir el puerto en el firewall.
Para fijar el puerto tenemos que modificar el archivo /etc/default/nfs-kernel-server

$ sudo nano /etc/default/nfs-kernel-server

Añadimos la siguiente línea

```
RPCMOUNTDOPTS="--port 2000"
```

```
# Number of servers to start up
# To disable nfsv4 on the server, specify '--no-nfs-version 4' here
RPCNFSDCOUNT=8

# Runtime priority of server (see nice(1))
RPCNFSDPRIORITY=0

# Options for rpc.mountd.
# If you have a port-based firewall, you might want to set up
# a fixed port here using the --port option. For more information,
# see rpc.mountd(8) or http://wiki.debian.org/?SecuringNFS
#RPCMOUNTDOPTS=--manage-gids
RPCMOUNTDOPTS="--port 2000"
```

Reiniciamos el servidor nfs con el comando antes explicado y comprobamos que todos los servicios están ligados a los puertos correspondientes.

$ rpcinfo -p

```
bd@ServerBD:~$ rpcinfo -p
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100024    1   udp  55207  status
    100024    1   tcp  33251  status
    100003    2   tcp   2049  nfs
    100003    3   tcp   2049  nfs
    100003    4   tcp   2049  nfs
    100227    2   tcp   2049
    100227    3   tcp   2049
    100003    2   udp   2049  nfs
    100003    3   udp   2049  nfs
    100003    4   udp   2049  nfs
    100227    2   udp   2049
    100227    3   udp   2049
    100021    1   udp  46340  nlockmgr
    100021    3   udp  46340  nlockmgr
    100021    4   udp  46340  nlockmgr
    100021    1   tcp  50634  nlockmgr
    100021    3   tcp  50634  nlockmgr
    100021    4   tcp  50634  nlockmgr
    100005    1   udp   2000  mountd
    100005    1   tcp   2000  mountd
    100005    2   udp   2000  mountd
    100005    2   tcp   2000  mountd
    100005    3   udp   2000  mountd
    100005    3   tcp   2000  mountd
```

Podemos ver que el servicio mountd ha quedado ligado al puerto 2000 tal y como habíamos indicado en el archivo de configuración.
Cuando se configure el firewall tendremos que abrir los siguientes puertos: 2049 nfs, 2000 mountd, 111 portmapper.

Ahora continuaremos con la configuración de la máquina cliente. Los siguientes pasos se realizarán en el webserver.

$ sudo apt-get install nfs-common

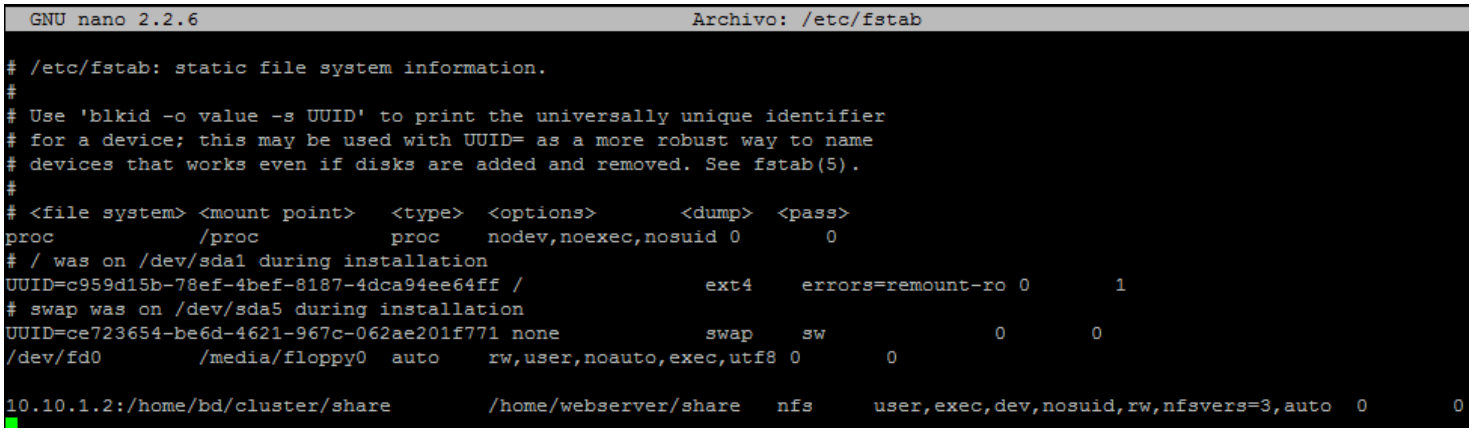
Creamos la carpeta donde se montará el recurso compartido.

```
$ mkdir /home/webserver/share
```

Editamos el archivo /etc/fstab para que la carpeta compartida share se cargue al arrancar la máquina.

```
$ sudo nano /etc/fstab
```

```
10.10.1.2:/home/bd/cluster/share        /home/webserver/share   nfs
user,exec,dev,nosuid,rw,nfsvers=3,auto  0       0
```

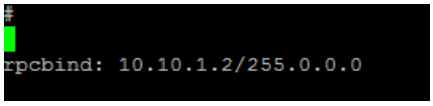


Debemos indicar que se trata de un recurso de tipo nfs y la versión de nfs que vamos a utilizar. Para el clúster se utiliza la versión nfserver=3.

El último paso es escribir la siguiente línea en el archivo /etc/hosts.allow

```
$ sudo nano /etc/hosts.allow
```

```
rpcbind: 10.10.1.2/255.0.0.0
```



Para comprobar que todo funciona correctamente podemos probar a remontar todas las unidades con el comando

```
$ sudo mount -a
```

Si no da ningún error ya hemos finalizado la instalación y configuración del sistema NFS. Cada vez que se arranque nuestro servidor web se cargará el recurso compartido del servidor bd.

7.8 Instalación SSH

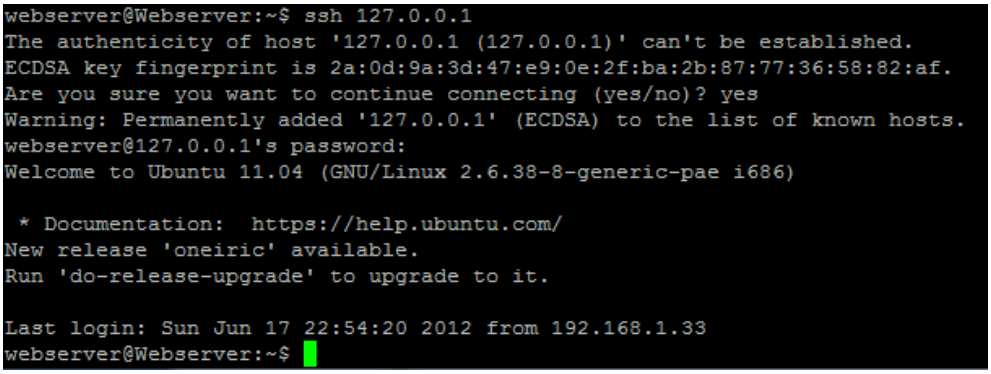
La instalación de los clientes ssh para conexiones remotas seguras se realiza con el siguiente comando

$ sudo apt-get install openssh-server

Ya que daremos acceso a los servidores para que se pueda conectar desde el exterior instalamos el ssh en los tres.

Para comprobar que se ha instalado correctamente escribimos

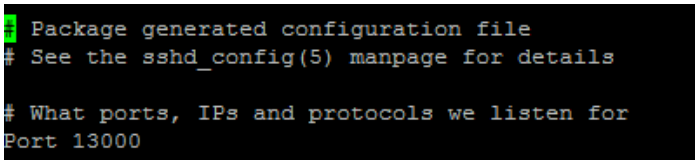
$ ssh 127.0.01



Para mayor seguridad en el servidor que hace de firewall modificamos el puerto del servicio ssh para que funcione con el puerto 13000. Debemos modificar el archivo /etc/ssh/sshd_config y cambiar el puerto que por defecto es el 22.

$ sudo nano /etc/ssh/sshd_config

Port 13000



También modificamos el archivo /etc/services

$ sudo nano /etc/services

Cambiamos los dos puertos tanto el tcp como el udp

ssh 13000/tcp
ssh 13000/udp

```
ftp             21/tcp
fsp             21/udp          fspd
ssh             13000/tcp                       # SSH Remote Login Protocol
ssh             13000/udp
telnet          23/tcp
```

El último paso es reiniciar el servicio

```
$ sudo /etc/init.d/ssh restart
```

7.9 Instalación Java

Para la instalación de java debemos descargarnos el paquete que corresponda a nuestro sistema operativo. En esta instalación se ha descargado la última versión que ofrece la página de oracle. http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1637583.html

Una vez descargado el paquete de 32 bits lo descomprimimos

```
$ tar -xvf jdk-7ua-linux-x64.tar.gz
```

Lo copiaremos en la carpeta /usr/lib/jvm/jdk1.7.0

```
$ sudo mv jdk1.7.0_04/ /usr/lib/jvm/jdk1.7.0
```

Debemos actualizar el path para que cuando ejecutemos aplicaciones java coja por defecto la versión que hemos instalado
Con el siguiente comando vemos cuantas referencias a java hay.

```
$ sudo update-alternatives --config java
```

```
update-alternatives: error: no hay alternativas para java.
```

Nos indica que no hay entradas para java y pasamos a instalarlo.

```
$ sudo update-alternatives --install /usr/bin/java java
/usr/lib/jvm/jdk1.7.0/jre/bin/java 0
```

El 0 indica la posición dentro de las entradas. En este caso no como no hay entradas la posición es 0.

Otra alternativa para instalar java es utilizando los repositorios.

```
$ sudo apt-get install openjdk-6-jdk
```

Para ver que se ha instalado correctamente ejecutamos

```
$ java -version
```

```
bd@ServerBD:~/scripts_BD$ java -version
java version "1.6.0_22"
OpenJDK Runtime Environment (IcedTea6 1.10.4) (6b22-1.10.4-0ubuntu1~11.04.1)
OpenJDK Client VM (build 20.0-b11, mixed mode, sharing)
bd@ServerBD:~/scripts_BD$
```

7.10 Instalación Iptables

En la actualidad iptables viene instalado por defecto con el sistema operativo Ubuntu. Si tuviésemos que instalarlo lo haríamos con la siguiente línea.

```
$ sudo apt-get install iptables
```

Iptables se configura línea a línea. Dado que esto es muy engorroso y la configuración para el firewall posee muchas líneas se ha hecho un script en el que guardar las reglas. Esto nos facilita las posibles modificaciones que se deseen hacer.

Creamos el script y escribimos las siguientes reglas.

```
$ nano iptables_firewall.sh
```

```
#!/bin/sh
##SCRIPT IPTABLES
echo -n "Reconfigurando Reglas iptables...\n"

##Variables
Ip_EXT="84.88.32.226"
Ip_BD="10.10.1.2"
Ip_WS="192.168.30.5"
eth_EXT="eth0"
eth_BD="eth1"
eth_WS="eth2"

##FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
echo "Flush de reglas"

#Politica por defecto ALL DENY
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
echo "Estableciendo política por defecto ALL DROP"
```

```
#Bit de Forward (imprescindible)
echo 1 > /proc/sys/net/ipv4/ip_forward

#Localhost total acceso
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

#Servicio web, acceso a la página web desde fuera
iptables -t nat -A PREROUTING -i $eth_EXT -p tcp --dport 80 -j DNAT --to $Ip_WS:80
iptables -A FORWARD -d $Ip_WS -p tcp --dport 80 -j ACCEPT

#ssh firewall por el Puerto 13000
iptables -A INPUT -p tcp --dport 13000 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 13000 -j ACCEPT

#ssh bd redigir Puerto 13001 del firewall al Puerto 22 del server BD
iptables -t nat -A PREROUTING -i $eth_EXT -p tcp --dport 13001 -j DNAT --to
$Ip_BD:22
iptables -A FORWARD -d $Ip_BD -p tcp --dport 22 -j ACCEPT

#ssh bd redirigir Puerto 13002 del firewall al Puerto 22 del server BD
iptables -t nat -A PREROUTING -i $eth_EXT -p tcp --dport 13002 -j DNAT --to
$Ip_WS:22
iptables -A FORWARD -d $Ip_WS -p tcp --dport 22 -j ACCEPT

#Navegacion desde firewall a webs(http) y webs seguras(https)
iptables -A INPUT -p tcp -m tcp --sport 80 -m state --state RELATED,ESTABLISHED -j
ACCEPT
iptables -A OUTPUT -p tcp -m tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp -m tcp --sport 443 -m state --state RELATED,ESTABLISHED
-j ACCEPT
iptables -A OUTPUT -p tcp -m tcp --dport 443 -j ACCEPT
#Config DNS
iptables -A INPUT -i $eth_EXT -p tcp -m tcp --sport 53 -j ACCEPT
iptables -A OUTPUT -o $eth_EXT -p tcp -m tcp --dport 53 -j ACCEPT
iptables -A INPUT -i $eth_EXT -p udp -m udp --sport 53 -j ACCEPT
iptables -A OUTPUT -o $eth_EXT -p udp -m udp --dport 53 -j ACCEPT

#Salida del servidorBD al exterior navegacion
iptables -t nat -A POSTROUTING -o $eth_EXT -s $Ip_BD -j MASQUERADE
iptables -A FORWARD -i $eth_BD -o $eth_EXT -s $Ip_BD -p tcp --dport 80 -j
ACCEPT
iptables -A FORWARD -i $eth_EXT -o $eth_BD -d $Ip_BD -p tcp --sport 80 -m state --
state RELATED,ESTABLISHED -j ACCEPT
#DNS
iptables -A FORWARD -i $eth_BD -o $eth_EXT -p tcp --dport 53 -j ACCEPT
iptables -A FORWARD -i $eth_EXT -o $eth_BD -p tcp --sport 53 -m state --state
RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_EXT -p udp --dport 53 -j ACCEPT
iptables -A FORWARD -i $eth_EXT -o $eth_BD -p udp --sport 53 -m state --state
RELATED,ESTABLISHED -j ACCEPT
```

```
#Salida del Webserver al exterior navegacion
iptables -t nat -A POSTROUTING -o $eth_EXT -s $Ip_WS -j MASQUERADE
iptables -A FORWARD -i $eth_WS -o $eth_EXT -s $Ip_WS -p tcp --dport 80 -j
ACCEPT
iptables -A FORWARD -i $eth_EXT -o $eth_WS -d $Ip_WS -p tcp --sport 80 -m state --
state RELATED,ESTABLISHED -j ACCEPT
#DNS
iptables -A FORWARD -i $eth_WS -o $eth_EXT -p tcp --dport 53 -j ACCEPT
iptables -A FORWARD -i $eth_EXT -o $eth_WS -p tcp --sport 53 -m state --state
RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_EXT -p udp --dport 53 -j ACCEPT
iptables -A FORWARD -i $eth_EXT -o $eth_WS -p udp --sport 53 -m state --state
RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_EXT -p tcp --dport 80 -j ACCEPT
iptables -A FORWARD -i $eth_EXT -o $eth_WS -p tcp --sport 80 -m state --state
RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_EXT -p udp --dport 80 -j ACCEPT
iptables -A FORWARD -i $eth_EXT -o $eth_WS -p udp --sport 80 -m state --state
RELATED,ESTABLISHED -j ACCEPT
#iptables -A FORWARD -i $eth_WS -o $eth_EXT -p tcp --dport 443 -j ACCEPT
#iptables -A FORWARD -i $eth_EXT -o $eth_WS -p tcp --sport 443 -m state --state
RELATED,ESTABLISHED -j ACCEPT
#iptables -A FORWARD -i $eth_WS -o $eth_EXT -p udp --dport 443 -j ACCEPT
#iptables -A FORWARD -i $eth_EXT -o $eth_WS -p udp --sport 443 -m state --state
RELATED,ESTABLISHED -j ACCEPT

#Sendmail
#iptables -A FORWARD -i $eth_WS -o $eth_EXT -s $Ip_WS -p tcp --dport 25 -j
ACCEPT # enviar correo
#iptables -A FORWARD -i $eth_EXT -o $eth_WS -d $Ip_WS -p tcp --sport 25 -m state -
-state RELATED,ESTABLISHED -j ACCEPT #recibir correo por el puerto 25

#Acceso NFS
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p tcp --dport 2049 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p tcp --sport 2049 -j
ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p udp --dport 2049 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p udp --sport 2049 -j
ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p tcp --dport 111 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p tcp --sport 111 -j
ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p udp --dport 111 -j
ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p udp --sport 111 -j
ACCEPT
```

```
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p tcp --dport 2000 -j ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p tcp --sport 2000 -j ACCEPT
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p udp --dport 2000 -j ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p udp --sport 2000 -j ACCEPT

#Acceso de la DMZ a la BD
iptables -A FORWARD -i $eth_WS -o $eth_BD -s $Ip_WS -p tcp --dport 3306 -j ACCEPT
iptables -A FORWARD -i $eth_BD -o $eth_WS -d $Ip_WS -p tcp --sport 3306 -j ACCEPT
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT


#Permitir hacer ping al webserver y servidor bd, desde el exterior no se puede hacer ping
iptables -A INPUT -i $eth_WS -p ICMP -j ACCEPT
iptables -A OUTPUT -o $eth_WS -p ICMP -j ACCEPT
iptables -A INPUT -i $eth_BD -p ICMP -j ACCEPT
iptables -A OUTPUT -o $eth_BD -p ICMP -j ACCEPT


echo "Configuración cargada con exito!"
```

Para ejecutar el script le damos permisos de ejecución

```
$ chmod +x iptables_firewall.sh
```

Y lo ejecutamos

```
$ sudo ./ iptables_firewall.sh
```

```
firewall@firewall:~$ sudo ./iptables_firewall.sh
Reconfigurando Reglas iptables...
Flush de reglas
Estableciendo polÃ-tica por defecto ALL DROP
ConfiguraciÃ³n cargada con exito!
firewall@firewall:~$
```

Sólo nos queda ejecutar automáticamente el script al iniciar la máquina.
Copiamos el script en la carpeta /etc/init.d

```
$ sudo cp iptables_firewall.sh /etc/init.d
```

Y editamos el archivo /etc/rc.local

$ sudo nano /etc/rc.local

./etc/init.d/iptables_firewall.sh

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

./etc/init.d/iptables_firewall.sh
exit 0
```

Cada vez que se reinicie el firewall se cargará el script y se mantendrán las reglas que hemos configurado.

7.11 Instalación Manager, Agents y PendingInfo

Las aplicaciones encargadas de gestionar el cluster están programadas en java. El proceso de instalación se puede diferenciar en dos partes

Compilación de los sources.
Configuración de carpetas, paths y usuarios.

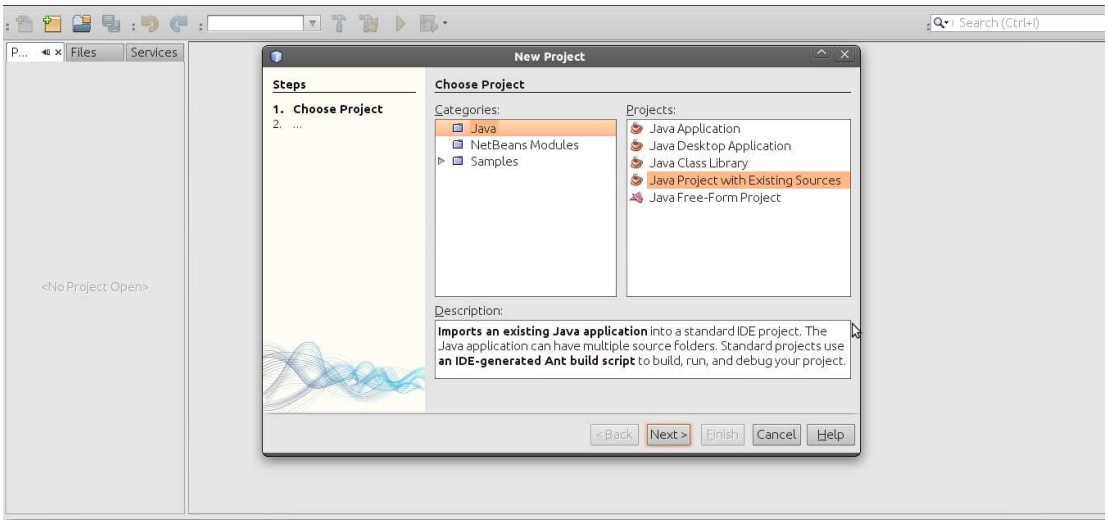
7.11.1 Compilación de los sources:

Necesitamos un entorno de desarrollo para poder compilarlas y se ha instalado el entorno Netbeans. Podemos instalarlo a partir del centro de software de Ubuntu o bien descargarlo de la página http://netbeans.org/downloads/.

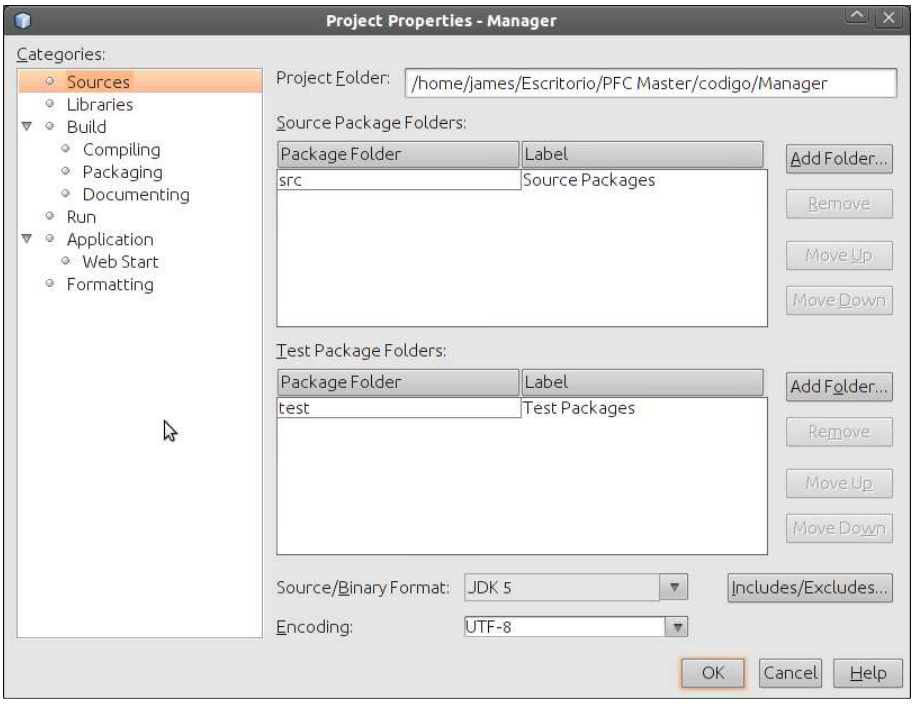
Nos descargamos el script de instalación y lo movemos a la carpeta donde vamos a instalarlo.

```
$ sudo mv netbeans-6.9.1-ml-linux.sh /usr/local/netbeans-6.9.1
$ sudo chmod +x netbeans-6.9.1-ml-linux.sh
$ sudo ./ netbeans-6.9.1-ml-linux.sh
```

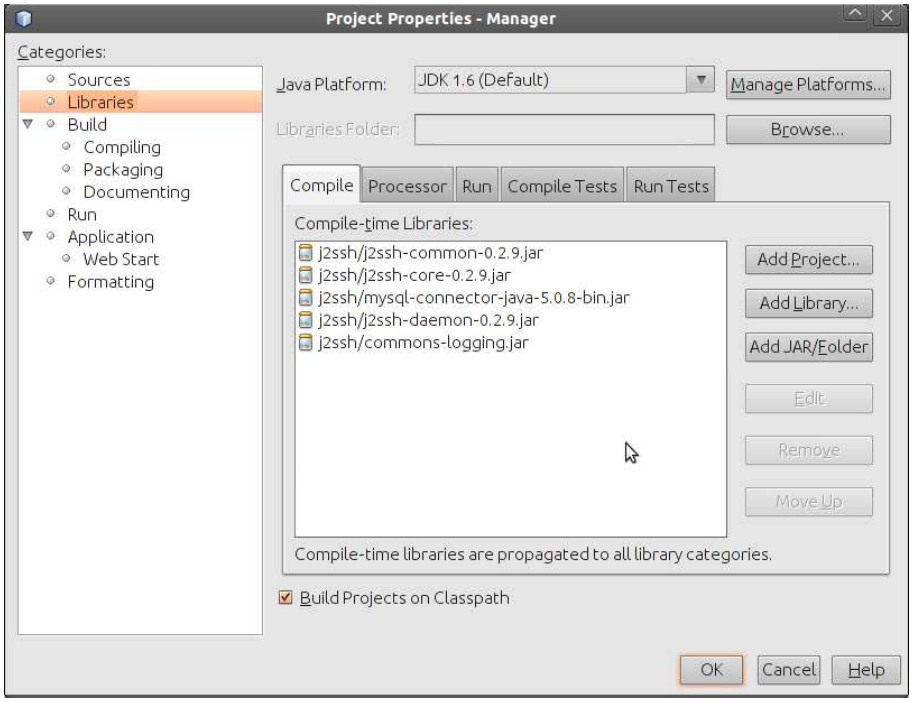
Ahora que está instalado creamos los proyectos para compilarlos. Abrimos netbeans y seleccionamos la opción de crear un proyecto nuevo a partir de código existente.



Indicamos donde se encuentran los sources del proyecto seleccionando los paths.

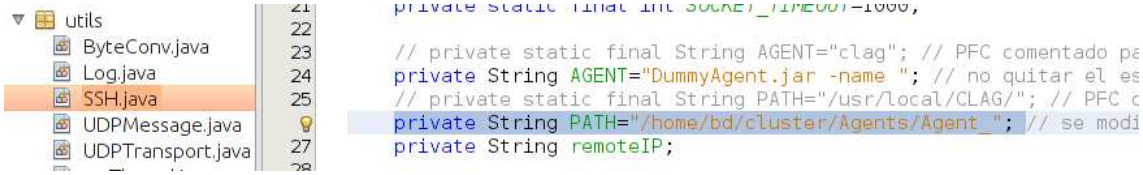


Agregamos las librerías que necesitemos para poder compilar el proyecto si es necesario. Para poder hacerlo seleccionamos el proyecto y hacemos click derecho sobre él. Seleccionamos la opción de propiedades y nos aparece la siguiente ventana. Nos desplazamos hasta la sección de librerías.



En el caso del manager antes de poder compilar debemos asegurarnos de que el path de los agents es el correcto e indicar que usuario de ssh se va a utilizar.

Accedemos al archivo SSH.java y modificamos la ruta donde se encuentran los agents.



También debemos indicar el usuario ssh con el que se conectarán los agents.

Generamos el proyecto compilándolo. Hay que indicar cual es la main-class del proyecto ejecutándolo una vez.

7.11.2 Configuración de carpetas, paths y usuarios.

Tanto el manager como los agents necesitan que les asignemos una IP para ello vamos a utilizar IPs virtuales.

Modificamos el archivo de interfaces para agregar las nuevas IPs

```
$ sudo nano /etc/network/interfaces
```

Añadimos las nuevas interfaces virtuales.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 10.10.1.2
netmask 255.0.0.0
gateway 10.10.1.5

# Ip manager
auto eth0:0
iface eth0:0 inet static
address 192.168.0.1
```

```
netmask 255.255.255.0

# Ip agent 10
auto eth0:1
iface eth0:1 inet static
address 192.168.0.10
netmask 255.255.255.0

# Ip agent 20
auto eth0:2
iface eth0:2 inet static
address 192.168.0.20
netmask 255.255.255.0

# Ip manager
auto eth0:3
iface eth0:3 inet static
address 192.168.0.30
netmask 255.255.255.0
```

Reiniciamos el servicio de red.

```
$ sudo /etc/init.d/networking restart
```

Creamos las carpetas donde vamos a poner las aplicaciones. Seguimos la siguiente estructura:

/home/bd/cluster/share
/home/bd/cluster/Manager
/home/bd/cluster/Agents
/home/bd/cluster/PendingInfo

En las carpetas Manager, PendingInfo pondremos los respectivos proyectos compilados. (Los .jar generados en la carpeta dist) y la carpeta de configuración de cada uno. En el manager copiamos también la carpeta lib que se encuentra junto con el ejecutable .jar en la carpeta dist.

En la carpeta Agents debemos crear una carpeta por cada agent del que disponga el sistema. Los nombres de cada carpeta vienen dados por la estructura de 'Agent_IP_virtual'

Dentro de la carpeta de cada agent copiaremos el .jar del proyecto DummyAgent y la carpeta de configuración.

Los archivos de configuración deben quedar de la siguiente forma.

Manager.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<Configuration>>
    <Manager IP="192.168.0.1" port="3001"></Manager>
    <agentPort>3002</agentPort>
    <dbName>ClusterMngr</dbName>
        <dbHost>localhost</dbHost>
        <dbPort>3306</dbPort>
        <dbUser>root</dbUser>
        <dbPasswd></dbPasswd>
    <logDir>.</logDir>
    <logName>ClusterManager.txt</logName>
    <helloInterval>100000</helloInterval>
    <maxLostHellos>3</maxLostHellos>
</Configuration>
```

Por cada agent un archivo de configuración dentro de su respectiva carpeta.

Agent.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<Configuration>
    <Agent id="10" IP="192.168.0.10" port="3002" sendProgressInterval="1000"></Agent>
    <Manager IP="192.168.0.1" port="3001"></Manager>
    <logDir>logDir</logDir>
    <logName>ClusterAgent.txt</logName>
    <helloInterval>60</helloInterval>
    <maxLostHellos>3</maxLostHellos>
</Configuration>
```

El último paso es crear un usuario para que el manager se pueda conectar a través de ssh a los agents.

```
$ sudo useradd alumne -p sistemes -d alumne
```

Para probar que el manager funciona correctamente accedemos a la carpeta donde se encuentra el ejecutable .jar y lo arrancamos de la siguiente forma.

```
$ java -jar Manager.jar
```

7.12 Listado de usuarios y contraseñas del sistema

Webserver
User: webserver Pass: webserver

Servidor BBDD
User: bd Pass: bd
User MySql: root Pass:root
User MySql web: cluster Pass: cluster
User ssh manager: alumne Pass: sistemes

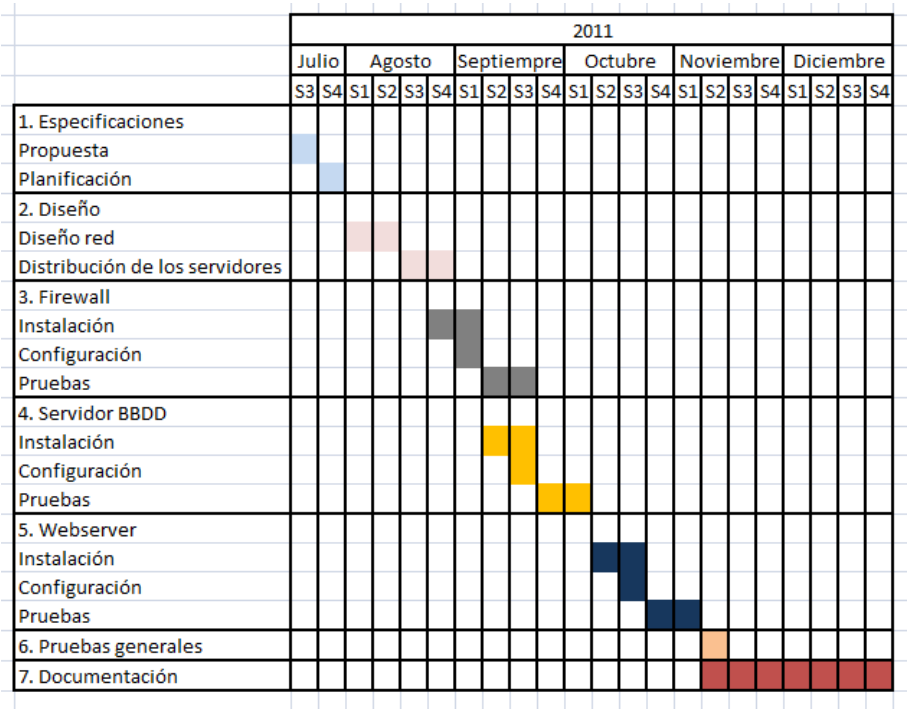
Firewall
User: firewall Pass: firewall

8. Gestión del proyecto

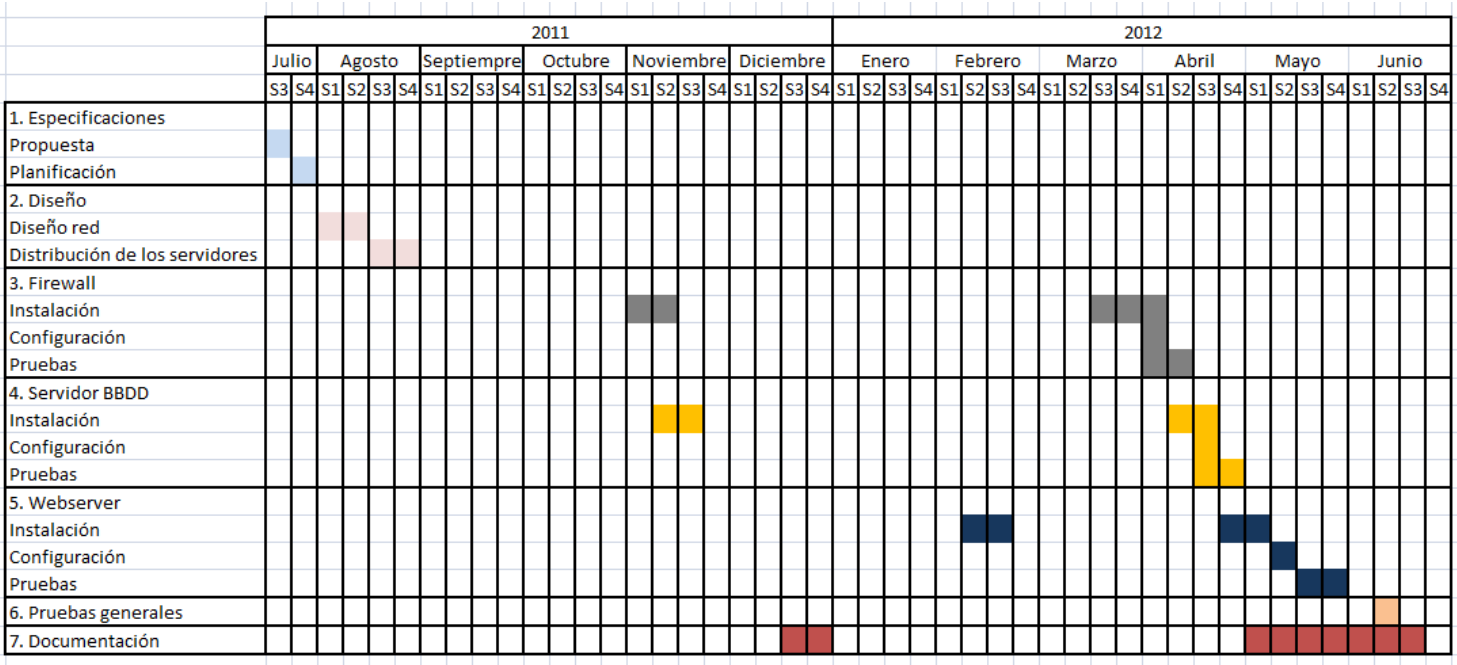
8.1 Planificación

En este apartado se muestra la planificación del proyecto así como la definición de las actividades y el presupuesto económico.

La planificación inicial fue la que se ve en la siguiente imagen.



Por diferentes motivos la planificación real es la que se ve a continuación.



Por motivos ajenos al proyecto los tiempos se han demorado tal y como se ve en el diagrama anterior. Algunas de las tareas han durado más tiempo del que se había previsto y los plazos no se han cumplido.

A continuación se detalla el número de horas invertido en cada tarea.

	Horas
Especificaciones	30
Diseño	26
Instalación	107
Configuración	135
Pruebas	65
Documentación	93

8.2 Valoración económica

Los costes del proyecto contemplan las horas de diseño, instalación y configuración de la red. Dado que el sistema se ha montado con equipo proporcionado por la universidad no se ha tenido en cuenta a la hora de realizar la valoración económica. Los componentes de hardware quedan fuera de esta valoración.

Actividad	Horas	Precio / Hora	Total
Analista	30	25€	750€
Diseñador	26	18€	468€
Administrador sist.	307	18€	5526€
			6744€

Para realizar la valoración económica no se han tenido en cuenta las horas dedicadas a documentar.

El precio total del proyecto es de 6744€, aplicando el 18% de IVA asciende a **7957,92€**.

9. Conclusiones

En la actualidad la virtualización (*es la creación -a través de software- de una versión virtual de algún recurso tecnológico*) está comiendo gran parte del terreno al diseño de redes físicas. La comodidad de no tener que montar máquinas reales y las ventajas que nos aporta el poder minimizar tiempos de instalación hacen de la virtualización de servicios una de las alternativas más utilizadas.

Después de desarrollar el proyecto de instalación y configuración de un cluster de computación se puede asegurar que incluso proyectos simples pueden acabar complicándose bastante.

Una buena planificación de proyecto es vital para poder llevarlo a cabo en los tiempos deseados. Por desgracia en muchas ocasiones imprevistos con los que no se habían contado pueden hacer que esa planificación se vea alterada y no se pueda tener el proyecto en las fechas marcadas.
Los problemas pueden venir de muy diversas formas. En el caso de éste proyecto han sido causas ajenas al trabajo académico que han hecho que el proyecto se demorase durante un largo periodo.

Otra conclusión que se puede extraer después de hablar con mucha gente es que en nuestra sociedad actual se suele caer en el error de no valorar en exceso el trabajo de los administradores de redes. En muchas ocasiones se suele quitar el merito que lleva el desarrollo de su trabajo. Se puede decir que es un trabajo que está a la sombra. Las cosas deben funcionar sin tener en cuenta que conlleva poder realizarlo. Se suele pensar que la instalación de una red es algo simple que tan solo consiste en conectar unas cuantas máquinas para que se puedan comunicar entre ellas. La realidad es que detrás de este montaje existen muchas invertidas y muchos problemas resueltos.

Otro tipo de problemas que pueden surgir y que si tienen que ver con el proyecto son de carácter técnico donde problemas con software o hardware hacen que se tenga que invertir más horas.

Por otra parte también se pueden extraer muchos otros aspectos positivos. El aumento de conocimiento y el aprender a moverse cuando hay desinformación nos da herramientas para poder seguir progresando como profesionales.

Por último quiero agradecer a todas las personas que me han apoyado y me han ayudado a poder realizar el proyecto. También agradecer el trabajo de mi tutor Luís Velasco.

10. Bibliografía

- http://www.wikipedia.org/
- http://www.ubuntu.com/
- http://www.apache.org/
- http://www.mysql.com/
- http://www.php.net/
- http://www.oracle.com/
- http://netbeans.org/
- http://www.pello.info/filez/firewall/iptables.html
- http://www.cyberciti.biz/faq/linux-detect-port-scan-attacks/